Aspects de la classification

Jean-François Mari, Amedeo Napoli

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Aspects de la classification

Jean-François Mari et Amedeo Napoli éditeurs

N■ 2909

Juin 1996

THÈME 3

ISSN 0249-6399

INRIA
LORRAINE

Aspects de la classification

Jean-François Mari et Amedeo Napoli éditeurs

Thème 3 — Interaction homme-machine,
images, données, connaissances
Projet SYCO

Rapport de recherche n° 2909 — Juin 1996 — 97 pages

Résumé : Les techniques de classification numérique ont toujours été présentes en reconnaissance des formes. Les réseaux de neurones montrent chaque jour leurs (très?) bonnes propriétés de classification, et la classification se fait de plus en plus présente en représentation des connaissances. Ainsi, ce rapport présente, simplement dans un but introductif, les aspects mathématiques, statistiques, neuromimétiques et cognitifs de la classification.

Mots-clé : catégorisation, classification, interprétation de connaissances

(Abstract: pto)

Actes de la journée *Classification* organisée par J.-F. Mari et A. Napoli à l'INRA LIAB Champenoux

Unité de recherche INRIA Lorraine
Technôpole de Nancy-Brabois, Campus scientifique,
615 rue de Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY (France)
Téléphone : (33) 83 59 30 30 – Télécopie : (33) 83 27 83 19
Antenne de Metz, technopôle de Metz 2000, 4 rue Marconi, 55070 METZ
Téléphone : (33) 87 20 35 00 – Télécopie : (33) 87 76 39 77

Some Aspects of the Classification

Abstract: Classification is an important issue in pattern recognition and knowledge representation. This report presents various basic aspects of classification: clustering by means of statistical technics or artificial neural networks, hierarchical knowledge representation and data mining. The issues of interpretation of concepts associated to the classes are also addressed.

Key-words: clustering, classification, knowledge representations

Préface

Le rapport *Aspects de la classification* est la trace écrite de la journée du même nom, organisé par Jean-François Mari et Amedeo Napoli, le 25 mars 1996, à l'INRA Champenoux, pour l'équipe RFIA du laboratoire CRIN–INRIA de Nancy, dirigée par Jean-Paul Haton.

L'idée de cette journée est née de l'air du temps, durant les précédentes journées annuelles de réflexion et de bilan de l'équipe RFIA, qui se sont déroulées en novembre 1995 à Metz. Cette journée fait suite — en quelque sorte — aux séminaires MAFIA : Mathématiques Appliquées et Fondamentales pour l'Intelligence Artificielle.Ceux-ci se proposent d'apporter des informations et des compléments sur un thème mathématique donné, qui entre dans les préoccupations de membres de l'équipe RFIA. La caractéristique des séminaires MAFIA a été conservée ici : les exposés sont introductifs et doivent servir de base au lecteur pour aborder le thème ou parfaire ses connaissances, et l'aider dans des investigations futures.

La journée *Aspects de la classification* se voulait donc une journée d'étude sur un thème qui préoccupe de nombreux membres, anciens et nouveaux, de l'équipe RFIA. A celà, plusieurs raisons peuvent être avancées. Les techniques de classification numérique ont toujours été présentes en reconnaissance des formes qui fut une des premières préoccupations de l'équipe. Les réseaux de neurones montrent chaque jour leurs (très ?) bonnes propriétés de classification, et enfin, la classification se fait de plus en plus présente en représentation des connaissances. Ainsi, les aspects mathématiques, statistiques, neuromimétiques et cognitifs ont pu être présentés.

Nous espérons que ce rapport technique servira également à toute personne désirant étudier et se documenter sur le thème de la classification, notamment grâce à l'ensemble des facettes proposées, mais aussi grâce aux nombreuses références associées à chaque article.

Pour terminer, les organisateurs de la journées voudraient remercier, les orateurs, pour le travail qu'ils ont fourni, que ce soit pour les exposés ou l'article associé, Florence le Ber, chercheur de l'équipe RFIA, qui nous a accueillis au milieu de la forêt de Champenoux dans le centre INRA dont elle dépend, l'ensemble des membres de l'équipe, qui a su se montrer un public très attentif, et Jean-Paul Haton, pour nous avoir donné la possibilité effective de mener à bien cette journée.

Avril 1996
Jean-François Mari
Amedeo Napoli

Programme

Lundi 25 mars 1996

9h–9h15	*Accueil*
9h15–10h	Jean-François Mari *Introduction à la classsification numérique*
10h–10h45	Amedeo Napoli *Classification et organisation hiérarchique des connaissances*
10h45–11h15	*Pause*
11h15–12h00	Frédéric Alexandre *Réseaux neuromimétiques et classification*
12h00–13h00	Alain Ketterlin *Classification et apprentissage : la formation de concepts structurés*
13h00–14h30	*Repas*
14h30–15h15	Jean-François Remm *Probabilités et réseaux de neurones*
15h15–16h15	Jean-Daniel Kant *Introduction au problème de l'interprétation des données en classification*
16h15–16h30	*Pause*
16h30–17h30	*Bilan et discussions*

Table des matières

Introduction à la classification numérique

Jean-François Mari
CRIN-CNRS & INRIA-Lorraine
C.-él. : jfmari@loria.fr

1.1 Classification et analyse discriminante

Effectuer une classification, c'est mettre en évidence des relations entre des objets et entre ces objets et leurs paramètres. A partir de mesures de proximités ou de dissemblances, il s'agit de construire une partition de l'ensemble des objets en un ensemble de *classes* les plus homogènes possible. Ce processus est aussi appelé *catégorisation*. Le cas le plus fréquent en reconnaissance des formes, et aussi le plus favorable, est sans conteste le cas où l'espace des mesures est $\mathbb{R}^p$ et où tous les paramètres sont quantitatifs comme « à combien de kilomètres de Tours habites tu? » par opposition aux paramètres qualitatifs comme « quelle appréciation mettre à cette très mauvaise copie? ». Nous retiendrons ce cas favorable dans la suite de cet exposé.

Il est fréquent que les résultats d'une classification ne reflètent pas les concepts usuels attachés aux objets. Ceci est dû à la distance choisie entre les objets. On dit souvent que la qualité d'une classification n'est pas que elle soit logique ou non, probable ou improbable mais que elle soit *profitable*. La classification permet principalement de trouver des régularités dans un grand tableau de nombres en dégageant des nuages de points homogènes auxquels on essaiera de donner un sens.

La classification et l'analyse discriminante vont souvent de pair. La première permet de dégager une partition à partir du tableau de données. La deuxième s'intéresse à expliquer une partition déjà existante, à l'aide de ce tableau de données. Un individu est décrit grâce à un vecteur de paramètres. En utilisant un échantillon de taille n issu de la population, l'analyse discriminante cherche à mettre à jour les relations linéaires qui définiront les frontières entre catégories. L'analyse discriminante a aussi un deuxième but qui est l'aide à la décision ; un nouvel individu arrive, dans quelle catégorie doit on le classer ?

Les paragraphes 2 à 6 introduisent les notions mathématiques nécessaires à la compréhension des deux types d'algorithmes de classification introduits aux paragraphes 7 et 8. Il s'agit des notions de covariance et d'inertie qui jouent des rôles importants en classification.

1.2 Éléments descriptifs d'un tableau de données quantitatives

1.2.1 Notations

On représente par un tableau X à n lignes et p colonnes les valeurs que prennent chacune des p variables sur les n individus. Toutes les variables sont quantitatives. Chaque ligne est un élément de $\mathbb{R}^p$. Cet espace est appelé l'espace des individus.

$$X = \begin{pmatrix} x_1^1 & x_1^2 & \cdots & x_1^p \\ x_2^1 & x_2^2 & \cdots & x_2^p \\ \vdots & \vdots & \ddots & \vdots \\ x_n^1 & x_n^2 & \cdots & x_n^p \end{pmatrix} = \left(\{x_i^j\}\right) \ , \ i = 1, \cdots n \ et \ j = 1, \cdots p$$

A chaque individu w_i est associée la i^e ligne du tableau définie par le vecteur :

$$\underline{x}_i = \begin{pmatrix} x_i^1 \\ x_i^2 \\ x_i^3 \\ \vdots \\ x_i^p \end{pmatrix}$$

L'ensemble des lignes constitue le nuage des individus. On peut être amené à associer à chaque individu w_i un poids normalisé $p_i > 0$ tel que :

$$\sum_{i=1}^{n} p_i = 1$$

De même on introduit l'espace des variables : $\mathbb{R}^n$. A chaque variable v_j est associée la j^{e} colonne du tableau définie par le vecteur :

$$\underline{x}^j = \begin{pmatrix} x_1^j \\ x_2^j \\ x_3^j \\ \vdots \\ x_n^j \end{pmatrix}$$

1.3 Définitions dans l'espace des variables

Moyenne : $\overline{x}^j = \sum_{i=1}^n p_i x_i^j$

Variance : $var(\underline{x}^j) = \sum_{i=1}^n p_i (x_i^j - \overline{x}^j)^2 = \sum_{i=1}^n p_i (x_i^j)^2 - (\overline{x}^j)^2$

Écart-type : $s(\underline{x}^j) = \sqrt{var(\underline{x}^j)}$

Covariance : A un couple de variables correspondant aux vecteurs $\underline{x}^j$ et $\underline{x}^k$ on associe la covariance :

$$cov(\underline{x}^j, \underline{x}^k) = \sum_{i=1}^n p_i (x_i^j - \overline{x}^j)(x_i^k - \overline{x}^k) = \sum_{i=1}^n p_i x_i^j x_i^k - \overline{x}^j \overline{x}^k$$

L'ensemble des covariances des p variables prises deux à deux constitue la matrice symétrique à p lignes et p colonnes appelée matrice de covariance totale du tableau :

$$V = \left(\{ cov(\underline{x}^j, \underline{x}^k) \} \right) \ , \ j = 1, \cdots p \ et \ k = 1, \cdots p$$

Les p éléments de la diagonale sont les variances des p variables.

Il existe une interprétation géométrique à la covariance entre deux variables (ou deux colonnes) de X qui peut être définie en termes de produits scalaires de $\mathbb{R}^n$. Considérons deux vecteurs de $\mathbb{R}^n$:

$$\underline{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} \quad \text{et} \quad \underline{y} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{pmatrix}$$

Un produit scalaire peut être défini de la façon suivante :

$$< \underline{x}, \underline{y} > = \sum_{i=1}^n p_i x_i y_i$$

où en terme de produit de matrices :

$$< \underline{x}, \underline{y} >= \begin{pmatrix} x_1 & x_2 & \cdots & x_n \end{pmatrix} \begin{pmatrix} p_1 & 0 & \cdots & 0 \\ 0 & p_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & p_n \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{pmatrix} = \underline{x}^t D_p \underline{y}$$

Supposons à présent que chaque colonne de X est un n-échantillon d'une variable centrée (*i.e.* de moyenne nulle). On obtient cela par le changement de variable :

$$\tilde{x}_i^j = x_i^j - \overline{x}^j$$

Si on les colonnes $\underline{\tilde{x}}^j$ et la matrice $\tilde{X}$ alors la covariance $cov(\underline{x}^j, \underline{x}^k)$ s'écrit :

$$cov(\underline{x}^j, \underline{x}^k) =< \underline{\tilde{x}}^j, \underline{\tilde{x}}^k >$$

et la variance devient :

$$var(\underline{x}^j) =< \underline{\tilde{x}}^j, \underline{\tilde{x}}^j >= \|\underline{\tilde{x}}^j\|^2$$

Corrélation : La corrélation est définie comme le quotient :

$$cor(\underline{x}^j, \underline{x}^k) = \frac{cov(\underline{x}^j, \underline{x}^k)}{s(\underline{x}^j)s(\underline{x}^k)}$$

Elle est le quotient du produit scalaire des vecteurs $\underline{\tilde{x}}^j$ et $\underline{\tilde{x}}^k$ par leurs normes. C'est donc le *cosinus* de l'angle formé. On peut regrouper ces quantités dans une matrice appelée matrice des corrélations. Dans cette matrice les éléments de la diagonale sont égaux à 1.

1.4 Éléments descriptifs des individus

Dans l'espace $\mathbb{R}^p$ où chaque ligne représente un individu, on peut considérer le nuage des individus. Commençons par représenter la distance entre deux individus.

1.4.1 Choix d'une distance entre individus

Dans $\mathbb{R}^p$ on utilise fréquemment des distances définies à partir de matrices (p,p) symétriques définies positives qui généralisent la distance euclidienne :

$$d_M(\underline{x},\underline{y}) = (\underline{x}-\underline{y})^t M(\underline{x}-\underline{y}), \text{ avec } \underline{x} \text{ et } \underline{y} \in \mathbb{R}^p$$

Si M est la matrice identité, d_M est le carré de la distance euclidienne.

1.4.2 Caractéristiques d'un nuage

centre de gravité : Le centre de gravité est le barycentre des n points considérés comme éléments de $\mathbb{R}^p$ affectés de leurs poids p_i.

$$\underline{\overline{x}} = \sum_{i=1}^{n} p_i \underline{x}_i = \begin{pmatrix} \sum_{i=1}^n p_i x_i^1 \\ \sum_{i=1}^n p_i x_i^2 \\ \vdots \\ \sum_{i=1}^n p_i x_i^p \end{pmatrix} = \begin{pmatrix} \overline{x}^1 \\ \overline{x}^2 \\ \vdots \\ \overline{x}^p \end{pmatrix}$$

Inertie : L'inertie par rapport à un point $\underline{x}$ et relativement à une distance d_M se définit de la façon suivante :

$$I_x = \sum_{i=1}^{n} p_i d_M(\underline{x_i},\underline{x})$$

Si $\underline{x}$ est le centre de gravité, on note $I_x = I$.

Remarque 1
Le centre de gravité du nuage est le point $\underline{\overline{x}}$ qui minimise :

$$I(x) = I_x = \sum_{i=1}^{n} p_i d_M(\underline{x_i},\underline{x})$$

Démonstration 1
On calcule le gradient de $I(x)$ par rapport à $\underline{x}$.

Sachant que si on appelle v un vecteur de $\mathbb{R}^p$, le gradient par rapport à v de la quantité :

$$f(v) = v^t M v$$

peut s'écrire :

$$\frac{\partial f(v)}{\partial v} = 2Mv$$

La recherche de l'extrémum se fait en annulant les dérivées partielles :

$$\frac{\partial I(x)}{\partial x} = 2M \sum_{i=1}^{n} p_i(\underline{x} - \underline{x}_i) = \underline{0}$$

donc :

$$\sum_{i=1}^{n} p_i(\underline{x} - \underline{x}_i) = \underline{0}$$

Cette dernière équation montre que $\underline{x}$ est le centre de gravité des $\underline{x_i}$.

Mesure de cohésion d'un nuage : A l'aide de la distance d_M, on définit une mesure de la cohésion du nuage par la quantité :

$$\mu(M) = \sum_{i=1}^{n} \sum_{j=1}^{n} p_i p_j d_M(\underline{x_i}, \underline{x_j})$$

On montre facilement que :

$$\mu(M) = 2I$$

Remarque 2
Pour un nuage d'individus donné, parmi les matrices M définissant une distance et dont le déterminant est égal à 1, celle qui minimise $\mu(M)$ est égale à :

$$(detV)^{\frac{1}{p}} V^{-1}$$

Démonstration 2
Se reporter à [Did 82] pages 330 – 332

Variance : Pour un nuage d'individus de poids quelconque, on définit la variance du nuage :

$var(X) = \frac{1}{\sum_{i=1}^{n} p_i} I$

1.5 Eléments descriptifs des nuages

Quand le nuage d'individus E représenté par X possède une partition en k classes $E_1, E_2, \cdots, E_k$, les centres de gravité, inertie, et matrices de covariance des sous nuages E_i sont reliés par des relations qui jouent un rôle important en classification automatique.

Centre de gravité des nuages

Soit $E_1, E_2, \cdots, E_k$ une partition de E. On a :

$$\begin{array}{lcccccc} E = & \{ & \overbrace{e_1, e_2, \cdots, e_i,}^{E_1} & \overbrace{\cdots, e_l,}^{E_2} & , \cdots & \overbrace{\cdots, e_n}^{E_k} & \} \\ 1 = & & \underbrace{p_1 + p_2 \cdots p_i +}_{q_1} & \underbrace{\cdots + p_l}_{q_2} + & , \cdots & \underbrace{\cdots + p_n}_{q_k} & \end{array}$$

On pose :

$$q_i = \sum_{e_i \in E_i} p_i$$

On définit les centres de gravité des nuages E_j et de l'ensemble E par :

$$g_j = \frac{1}{q_j} \sum_i p_i e_i \quad \text{pour} \quad e_i \in E_j$$

$$\underline{\overline{x}} = \sum_{j=1}^{k} q_j g_j$$

Matrices de covariance interclasse et intraclasse

La matrice :

$$V_j = \frac{1}{q_j} \sum_{e_i \in E_j} p_i (e_i - g_j)(e_i - g_j)^t$$

est la matrice de covariance de E_j. Par définition, on appelle W (comme *within*) la matrice intraclasse définie comme la moyenne pondérée des V_j :

$$W = \sum_{j=1}^{k} q_j V_j$$

De même, on définit la matrice interclasse B (comme *between*) comme la matrice de covariance des centres de gravité g_j affectés de leur masse q_j :

$$B = \sum_{j=1}^{k} q_j (g_j - \overline{x})(g_j - \overline{x})^t$$

Cette matrice $(p \times p)$ n'est pas inversible car les k centres de gravité sont dans un sous-espace de dimension $k - 1 < p$. On montre facilement :

$$V = W + B$$

Inertie interclasse et inertie intraclasse

Le théorème de Koenig-Huygens (dont la démonstration se trouve dans [Did 82] pp. 59 – 61) établit la relation entre l'inertie totale et l'inertie de chaque classe par rapport à son centre de gravité :

$$I(E) = \sum_{j=1}^{k} I(E_j) + \sum_{j=1}^{k} q_j d_M(g_j, \overline{x})$$

1.6 Distances

1.6.1 Distances inter-groupes

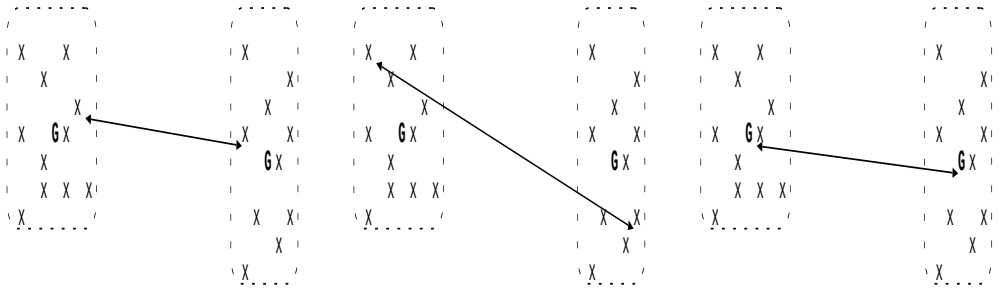
Les figures 1.1(a), 1.1(b) et 1.1(c) illustrent trois distances usuelles entre deux groupes d'individus.

1.6.2 Cas où deux classes contiennent des populations normales

Dans le cas où deux classes contiennent des populations normales, la divergence entre ces deux classes donne une mesure de leurs similarités. La notion de divergence est très clairement introduite dans l'ouvrage [Tou 74] aux pages 291 – 298.

Soient x un individu, w_i et w_j deux classes dans lesquelles x peut être classé. On note $p_i(x) = p(x/w_i)$ la probabilité d'apparition de x quand celui-ci appartient à la classe w_i. On définit de même $p_j(x) = p(x/w_j)$. L'information discrimante entre ces deux évènements se mesure par :

$$u_{ij} = \ln \frac{p_i(x)}{p_j(x)}$$



(a) lien minimum (b) lien maxi. (c) entre centres de gravité

FIG. 1.1 – *Distances entre groupes*

L'espérance de cette mesure pour la classe w_i s'écrit :

$$I(i,j) = \int_x p_i(x) \ln \frac{p_i(x)}{p_j(x)} dx$$

Pour avoir une mesure de ressemblance qui ne dépende pas de l'ordre des classes w_i et w_j, on pose :

$$Jij = I(i,j) + I(j,i)$$

Jij est la divergence entre les classes w_i et w_j. Cette quantité a une forme analytique quand les classes ont une répartition gaussienne :

$$p_i(x) = \mathcal{N}(\mathbf{x}; \mu_\mathbf{i}, \Sigma_\mathbf{i}) = \frac{1}{\sqrt{(2\pi)^p \det \Sigma_\mathbf{i}}} \exp^{-\frac{1}{2}(\mathbf{x}-\mu_\mathbf{i})^t \Sigma_\mathbf{i}^{-1} (\mathbf{x}-\mu_\mathbf{i})}$$

$$p_j(x) = \mathcal{N}(\mathbf{x}; \mu_\mathbf{j}, \Sigma_\mathbf{j}) = \frac{1}{\sqrt{(2\pi)^p \det \Sigma_\mathbf{j}}} \exp^{-\frac{1}{2}(\mathbf{x}-\mu_\mathbf{j})^t \Sigma_\mathbf{j}^{-1} (\mathbf{x}-\mu_\mathbf{j})}$$

Dans ce cas J_{ij} s'écrit :

$$J_{ij} = \frac{1}{2} tr[(\Sigma_i - \Sigma_j)(\Sigma_j{}^{-1} - \Sigma_i{}^{-1})] + \frac{1}{2} tr[(\Sigma_i{}^{-1} + \Sigma_j{}^{-1})(\mu_i - \mu_j)(\mu_i - \mu_j)^t]$$

Un cas particulier est intéressant :

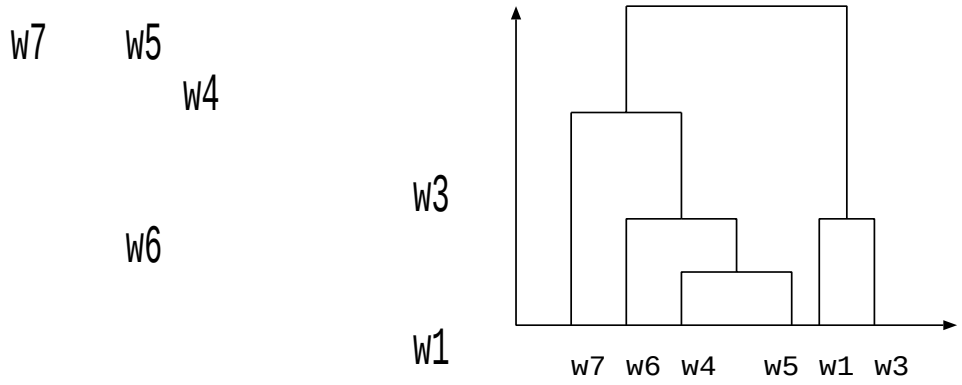
$$\Sigma_i = \Sigma_j = \Sigma$$

Dans ce cas J_{ij} devient la distance de Mahalanobis :

$$J_{ij} = (\mu_i - \mu_j)^t \Sigma^{-1} (\mu_i - \mu_j)$$

1.7 Classification par hiérarchie et par arbre

On cherche à représenter les points $w_1, w_2, \ldots, w_n$ par un ensemble de parties hiérarchiquement emboîtées. Par exemple, la hiérarchie indiquée figure 1.2(b) représente l'ensemble des points du plan donnés en figure 1.2(a) si on prend comme distance la distance euclidienne (d'après Diday [Did 82]).



(a) nuage de points (b) hiérarchie

FIG. 1.2 – *Classification hiérarchique*

1.7.1 Indices d'agrégation

La construction d'une hiérarchie nécessite la connaissance d'une mesure de ressemblance entre individus entre eux, entre groupes entre eux et entre groupe et individu. Cette mesure de ressemblance s'appelle aussi *indice d'agrégation*. Elle est

définie à partir des distances entre individus et des poids des parties de la hiérarchie. Parmi les plus utilisés, on peut citer :

- l'indice d'agrégation du lien minimum entre 2 parties h_1 et h_2 illustré par la figure 1.1(a) :
$$\delta(h_1, h_2) = \min_{w_i \in h_1,\ w_j \in h_2} d(w_i, w_j)$$

- l'indice d'agrégation du lien maximum illustré par la figure 1.1(b) :
$$\delta(h_1, h_2) = \max_{w_i \in h_1,\ w_j \in h_2} d(w_i, w_j)$$

- l'indice des moyennes des distances illustré par la figure 1.1(c) :
$$\delta(h_1, h_2) = \frac{1}{|h_1||h_2|} \sum_{w_i \in h_1,\ w_j \in h_2} d(w_i, w_j)$$
avec $|h_i|$ le cardinal de h_i

  Autrement dit, l'indice d'agrégation entre deux parties est la moyenne des distances entre les individus pris dans les deux classes.

- l'indice d'agrégation des centres de gravité :
$$\delta(h_1, h_2) = d(G(h_1), G(h_2))$$

- l'indice d'agrégation de la variation de l'inertie :
$$\delta(h_1, h_2) = \frac{p(h_1)p(h_2)}{p(h_1) + p(h_2)} d(G(h_1), G(h_2)) \quad (1.1)$$
avec $G(h_i)$ centre de gravité de h_i

  Cet indice est couramment employé. On a vu précédemment que l'inertie d'un nuage est proportionnelle à la cohésion du nuage. A l'aide de cet indice, on commence par l'agrégation des classes qui fera diminuer le moins l'inertie interclasse.

- l'indice d'agrégation de la variation de la variance :
$$\delta(h_1, h_2) = \frac{p(h_1)p(h_2)}{(p(h_1) + p(h_2))^2} d(G(h_1), G(h_2))$$

1.7.2 Algorithme

L'algorithme est simplement expliqué dans [Leb 82]. Il procède par regroupements successifs :

1. partir d'une partition dont les classes sont réduites à un élément ;
2. rechercher le couple de classes dont l'indice d'agrégation est minimum (par exemple : agrégation en fonction du lien minimum ou de la perte d'inertie inter-classe) ;
3. construire la classe résultant de la fusion des deux précédentes ;
4. recommencer la recherche jusqu'au regroupement final.

Cet algorithme est rapide. Au fur et à mesure des regroupements, le nombre d'itérations pour calculer les nouveaux indices d'agrégation diminue. Il permet de s'arrêter sur un nombre de classes donné *a priori* ou sur un critère calculé à partir de l'indice d'agrégation comme la perte d'inertie. Son principal inconvénient est son encombrement, car au départ, il faut stocker une matrice de distances (ou ressemblances) entre points qui, bien que symétrique, peut être encombrante.

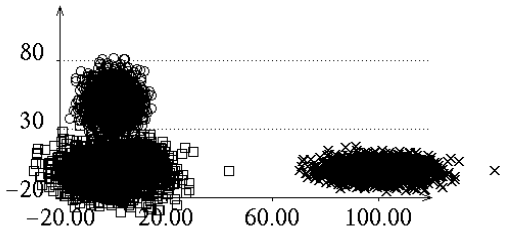
1.8 Classification descendante

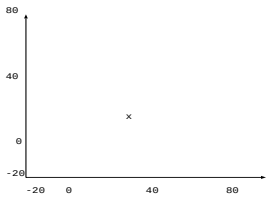
Une autre technique de classification, connue sous le nom de classification par nuées dynamiques [Did 82] procède par divisions successives du nuage initial. Chaque classe est représentée par un noyau qui peut être un point, un ensemble de points ou un sous-espace. Dans le cas particulier où le noyau est un point de $\mathbb{R}^p$, l'algorithme est connu sous le nom *algorithme des centres mobiles* en France et *K-means* outre atlantique. Un exemple est donnée à la figure 1.3.

1.8.1 Classification descendante par centres mobiles

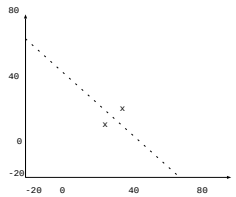
L'ensemble de points à partitionner est représenté par la figure 1.3-a. Il est constitué de 3 nuages obtenus par simulation dans le plan de lois normales.

L'algorithme commence à calculer le centre de gravité de l'ensemble (figure 1.3-b) puis « invente » deux points fictifs obtenus par une petite perturbation du centre de gravité (figure 1.3-c).

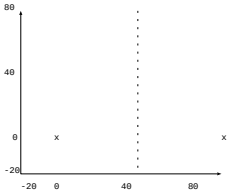




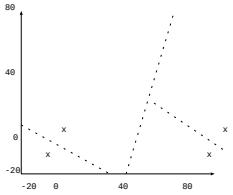
(a) moyenne générale



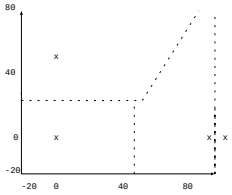
(b) perturbation



(c) partition opt. en 2 classes



(d) perturbations



(e) partition opt. en 4 classes

FIG. 1.3 – *Classification descendante par centres mobiles*

Ces deux points sont considérés comme les noyaux de deux classes. L'algorithme affecte ensuite l'ensemble des points à chacune de ces deux classes à l'aide d'un critère de distance : c'est la fonction d'affectation. Les deux classes définies, on calcule leurs centres de gravité : c'est la fonction de représentation. On parcourt à nouveau

l'ensemble des points pour les affecter à ces deux classes et on recalcule les nouveaux centres de gravité. On démontre que la variance intra-classe diminue à chaque nouvelle affectation [Sap 90]. La suite de partitions crée converge vite vers une partition où la variance intra-classe ne diminue plus. En général 3 itérations suffisent. On se retrouve donc avec une partition en deux classes (figure 1.3-d). L'algorithme fait subir la même division à chacune des classes (figure 1.3-e) et on obtient la division en 4 classes représentée par la figure 1.3-f. On voit que une classe a été subdivisée « à tort ». Pour éviter ce phénomène, on fait suivre une classification descendante par une phase de consolidation [Leb 95] obtenue par une classification hiérarchique dans laquelle les divisions outrancières sont corrigées. Du moins, l'espère-t'on !

Toutes les frontières entre les classes sont les droites médiatrices des segments joignant les centres de gravité. Leurs équations en x et y sont linéaires. Il existe d'autres méthodes de classification à l'aide de réseaux de neurones qui fournissent des frontières dont les équations sont polynomiales en x et y.

Contrairement à la classification ascendante dans laquelle l'algorithme donne toujours la meilleure partition pour le critère d'agglomération choisi, l'algorithme des centres mobiles peut donner des partitions différentes suivant les initialisations. La partition obtenue correspond à une minimisation locale de la variance intra-classe. Dans une classification ascendante, il peut exister une meilleure partition, mais l'algorithme ne peut pas la trouver et il faut envisager d'autres méthodes.

1.9 Pour en savoir plus

Différents ouvrages nous ont aidés dans la rédaction de cette présentation de quelques méthodes de classification numérique. L'ouvrage de E. Diday [Did 82] fut pendant longtemps le premier livre d'analyse de données complet. Il reste difficile à lire pour un néophyte à cause de ses notations compliquées. Un autre ouvrage écrit par L. Lebart [Leb 82] est plus simple. A la fin de l'ouvrage, on trouve des méthodes d'analyse numérique et des programmes intéressant l'analyse des données. C'est un bon livre pour l'informaticien qui débute en analyse de données. Une nouvelle édition est parue récemment [Leb 95]. Citons ensuite un ouvrage en langue anglaise [Tou 74] consacré à la reconnaissance des formes et qui traite de l'utilisation des probabilités et statistiques dans ce domaine de l'informatique. Cet ouvrage introduit les notions mathématiques pour la reconnaissance des formes. Chaque chapitre est indépendant de ses voisins et isole une notion. Pour terminer, il ne faut pas oublier la dernière édition de l'ouvrage de G. Saporta [Sap 90] qui contient un très bon résumé des méthodes de classification et d'analyse discriminante.

Bibliographie

[Did 82] E. Diday, J. Lemaire, J. Pouget, et F. Testu. *Éléments d'analyse de données.* Dunod, 1982.

[Leb 82] L. Lebart, A. Morineau, et J.P. Fenelon. *Traitement des données statistiques : Méthodes et programmes.* Dunod, 1982.

[Leb 95] L. Lebart, A. Morineau, et M. Pinon. *Statistique exploratoire multidimensionnelle.* Dunod, 1995.

[Sap 90] G. Saporta. *Théories et méthodes de la statistique.* Publications de l'institut français du pétrole, 1990.

[Tou 74] J. T. Tou et R. Gonzales. *Pattern Recognition Principles.* Addison-Wesley, 1974.

Classification et organisation hiérarchique des connaissances

Amedeo Napoli
CRIN-CNRS & INRIA-Lorraine
C.-él. : napoli@loria.fr

2.1 Introduction

Dans cet exposé, nous étudions certaines caractéristiques liées à l'organisation hiérarchique des bases de connaissances des systèmes d'intelligence artificielle. Nous nous intéressons essentiellement aux opérations de *catégorisation* et de *classification*. La catégorisation recouvre les processus de construction (automatique et manuelle) de *catégories* ou *classes*, ainsi que l'organisation de ces classes. La classification recouvre les processus de reconnaissance de la classe d'un objet ou individu et l'insertion d'une classe dans une hiérarchie, qui sont à la base du *raisonnement par classification*. Ce mode de raisonnement opère sur une hiérarchie de classes et permet d'expliquer ou de reconnaître un (nouvel) objet en identifiant ses caractéristiques, relativement à la hiérarchie étudiée.

Une classe `C` qui représente un concept du monde réel se définit comme une collection de propriétés structurelles et (éventuellement) fonctionnelles, ou procédurales, qui décrivent respectivement l'*état* de `C` et les *opérations* qui lui sont associées. Les classes sont organisées par l'intermédiaire d'une relation (d'ordre partiel) selon leur niveau d'abstraction en une *hiérarchie* qui est un graphe étiqueté orienté et sans cycle. Une classe `C` *partage* ou *hérite* les propriétés des classes qui sont plus générales qu'elle, les classes *ascendantes* de `C`, tandis que les propriétés de `C` sont héritées par les classes qui sont plus spécifiques que `C`, les classes *descendantes* de `C`. Le partage de propriété repose essentiellement sur la transitivité de la relation d'ordre partiel qui est à la base de la hiérarchie. Globalement, il est ainsi possible de définir une *représentation hiérarchique* par l'équation : *représentation hiérarchique = classes + ordre partiel + partage de propriétés*.

Du point de vue de l'implantation, les représentations à objets [Nap 92] et les logiques terminologiques [Neb 90a] sont des formalismes qui permettent de représenter naturellement un ensemble de concepts (du monde réel) sous la forme d'une hiérarchie d'objets sur laquelle opère le raisonnement par classification.

2.2 Aspects des représentations hiérarchiques

2.2.1 Décomposition canonique d'un préordre et hiérarchie indicée

Du point de vue mathématique [Bar 70], et plus particulièrement de l'analyse de données, l'étude des liens entre les relations d'équivalence et les relations d'ordre partiel montre comment se combinent les processus de formation et d'organisation des classes. Considérons un ensemble E et un préordre $\mathcal{R}$ (une relation réflexive et transitive)[1]. Soit la relation ρ définie par : x ρ y si et seulement si x $\mathcal{R}$ y et y $\mathcal{R}$ x. Elle est réflexive et transitive car $\mathcal{R}$ est un préordre, mais aussi symétrique. Par suite, ρ est une relation d'équivalence sur l'ensemble E. L'ensemble des classes d'équivalence est dénoté par le quotient E/ρ. Toute classe C dans E/ρ est considérée comme une classe qui regroupe des individus équivalents relativement à ρ. La formation de classes repose donc sur l'existence d'un préordre et de la relation d'équivalence associée.

Considérons maintenant l'organisation externe des classes appartenant à E/ρ, et la relation σ, définie sur E/ρ par : X σ Y si et seulement si $\exists$ x $\in$ X, $\exists$ y $\in$ Y tels que x $\mathcal{R}$ y. Deux classes d'équivalence X et Y sont en relation si au moins un élément de X est en relation, pour le préordre $\mathcal{R}$, avec un élément de Y. La relation σ est réflexive et transitive car $\mathcal{R}$ est un préordre, mais aussi antisymétrique : toutes les relations entre deux classes sont orientées dans le même sens, sinon les deux classes sont identiques. En effet, supposons que : $\exists$ x, x' $\in$ X, $\exists$ y, y' $\in$ Y, tels que x $\mathcal{R}$ y et y' $\mathcal{R}$ x' ; dans ce cas, y $\mathcal{R}$ y' car y et y' sont dans la même classe Y, x' $\mathcal{R}$ x car x' et x sont dans la même classe X, et par transitivité de $\mathcal{R}$, il vient que y $\mathcal{R}$ x, ce qui signifie que x et y sont dans la même classe. Par suite, la relation σ définit un ordre partiel sur l'espace quotient E/ρ, et elle permet d'organiser l'ensemble des classes en une hiérarchie : X a pour ascendant Y, noté X $\preceq$ Y, si et seulement si X σ Y.

Plus précisément, une *hiérarchie (indicée) de parties* se définit de la façon suivante [Did 82] :

Soit E un ensemble et H une famille de parties de E.

H est une *hiérarchie (de parties)* de E si :

(i) $\forall$ h, h' $\in$ H : h $\cap$ h' $\neq \emptyset \Longrightarrow$ h $\subseteq$ h' ou h' $\subseteq$ h

(autrement dit h $\cap$ h' $\subseteq$ {h, h', $\emptyset$}).

(i) $\forall$ h, h' $\in$ H : h $\cap$ h' $\subseteq$ {h, h', $\emptyset$}.

1. À partir d'une relation transitive notée $\mathcal{T}$, il est possible de définir une relation réflexive (et donc un préordre) en posant : x $\mathcal{R}$ y si et seulement si x $\mathcal{T}$ y ou x = y.

(ii) $\forall$ x $\in$ E : {x} $\in$ H.

(iii) E $\in$ H.

(iv) La hiérarchie est dite *indicée* s'il existe une fonction i : H $\longrightarrow \mathbf{R}^+$ telle que (hiérarchie indicée au sens large) :

$\forall$ x $\in$ E : i({x}) = 0,

$\forall$ h, h' $\in$ H, h $\subseteq$ h' $\Longrightarrow$ i(h) $\leq$ i(h') (divers choix sont possibles pour la fonction i, voir [Did 82]).

2.2.2 Hiérarchies conceptuelles

Une classe C représente un concept du monde réel. Elle possède une *identité* — un nom par l'intermédiaire duquel elle peut être manipulée — et un ensemble de propriétés correspondant aux caractéristiques du concept représenté. Une classe permet de décrire un ensemble d'entités individuelles ayant même structure et même comportement. Une *hiérarchie conceptuelle* $\mathcal{H} = (\mathcal{X}, \preceq, \omega)$ est un graphe orienté sans circuit, où $\mathcal{X}$ est un ensemble de classes (les sommets du graphe), $\preceq$ est une relation d'ordre partiel (les arcs du graphe sont déterminés par les relations entre classes), et ω est l'élément maximal de $\mathcal{X}$ suivant $\preceq$; ω est appelé la *racine* de la hiérarchie, et elle est supposée toujours exister.

Cette définition de hiérarchie conceptuelle est une adaptation de la notion de *graphe d'héritage* donnée dans [Duc 89] : un *graphe d'héritage* $G_H = (\mathcal{X}, \mathcal{H}, \omega)$ est un graphe orienté sans circuit muni d'une racine, où $\mathcal{X}$ est un ensemble d'objets, dont ω, la racine, est le maximum suivant la relation de spécialisation $\mathcal{H}$.

Du point de vue de l'implantation, une classe est une entité générique qui possède un nom et qui définit un *modèle* à partir duquel s'engendrent ses *instances*, qui représentent des entités individuelles. Un système de représentation où l'unité de connaissance est la classe (accompagnée de ses instances), où les classes sont organisées en hiérarchies, et où le processus de classification fait partie intégrante des opérations de raisonnement est appelé une *représentation à objets* [Nap 93] [Nap 92]. Les représentations à objets sont une extension des systèmes à héritage classiques — langages de classes, langages de frames et langages hybrides [Mas 89] — et entretiennent des relations étroites avec les *logiques terminologiques* ou *logiques de descriptions* [Neb 90a].

2.2.3 La catégorisation

La *catégorisation* est une opération qui consiste à découvrir des régularités pour regrouper des entités individuelles en classes :

$\{\mathtt{x_i}\}_{\mathtt{i}\in\mathtt{I}} \longrightarrow \{\mathtt{C_j}\}_{\mathtt{j}\in\mathtt{J}} \longrightarrow \mathcal{H} = (\mathcal{X}, \preceq, \omega)$.

Les entités individuelles $\mathtt{x_i}$ sont regroupées dans des classes $\mathtt{C_j}$ et l'ensemble des classes est organisé en une hiérarchie $\mathcal{H} = (\mathcal{X}, \preceq, \omega)$.

Pour une classe C donnée, l'*intension* de C fait référence à l'ensemble des propriétés de C, et l'*extension* de C à l'ensemble des instances de (engendrées par) C. Ainsi, si deux classes C et D sont telles que $\mathtt{D} \preceq \mathtt{C}$, alors (généralement) les deux expressions duales suivantes sont vérifiées :

$\mathtt{Intension(C)} \subseteq \mathtt{Intension(D)}$ et $\mathtt{Extension(D)} \subseteq \mathtt{Extension(C)}$.

Une distinction importante porte sur la structure interne d'une classe qui peut être *monothétique* ou *polythétique* [Ler 70] [Vig 91] :

- Une classe est *monothétique* si elle est caractérisée par une propriété : un individu appartient à la classe si et seulement si il possède la propriété.

- Une classe est *polythétique* si, étant donné un ensemble d'attributs $\mathtt{A} = \{\mathtt{a}_1, \mathtt{a}_2, \ldots, \mathtt{a}_p\}$:

  (i) chaque individu recouvert par la classe possède un « grand nombre » des attributs de A,

  (ii) chaque attribut de A est possédé par un « grand nombre » des individus recouvert par la classe,

  (iii) il n'existe pas (*a priori*) un attribut qui soit possédé par tous les individus recouverts par la classe.

La catégorisation peut s'envisager selon plusieurs modes [Nap 94a]. La *catégorisation conceptuelle* consiste à regrouper des objets en classes qui matérialisent des concepts du domaine étudié. Les objets sont discriminés sur la base de propriétés qu'ils partagent, mais aussi sur la base des connaissances disponibles sur le domaine. Ce mode de catégorisation est à rapprocher de la classification automatique pratiquée en analyse de données [Did 82].

La *catégorisation hiérarchique incrémentale*, appelée aussi *formation incrémentale de hiérarchies de concepts*, a le même but que la catégorisation conceptuelle. Toutefois, par opposition à un traitement global, les objets étudiés sont traités les uns après les autres, et l'ensemble des classes produit est obligatoirement une hiérarchie. Les informations associées aux objets traités permettent de définir des propriétés

pour les classes qui vont être formées et qui vont servir à expliquer les caractéristiques de ces objets en cours de traitement.

La conception d'une hiérarchie d'héritage est généralement une opération *manuelle* de catégorisation. Dans une hiérarchie d'héritage, les nouvelles classes *spécialisent* les classes existantes par adjonction de nouvelles propriétés et/ou par substitution de valeurs dans des propriétés déjà définies. Si la classe `B` spécialise la classe `A`, ce qui se note $\mathtt{B} \preceq_h \mathtt{A}$, alors $\mathtt{P_A} \subseteq \mathtt{P_B}$, où $\mathtt{P_A}$ dénote l'ensemble (des noms) des propriétés attachées à la classe `A`, l'intension de `A` en quelque sorte. Le mécanisme d'héritage permet d'exploiter la hiérarchie d'héritage et de résoudre le problème de partage de propriétés : étant donné une classe `C` et une propriété `p`, retrouver la valeur de `p` pour `C`.

Nous n'en parlons pas ici, mais il existe également différentes approches de la catégorisation qui sont relatives aux sciences cognitives, et qui sont présentées (entre autres) dans [Kle 90] et [Dub 91]. Dans le même ordre d'idées, la référence [Mec 93] est intéressante à plus d'un titre, car elle regroupe une collection de papiers montrant les liens existant entre analyse de données numériques, symboliques, représentation de connaissances et sciences cognitives.

2.2.4 La classification

Du point de vue cognitif, la *classification* est un processus guidé par le principe de *remémoration* : étant donné une hiérarchie de référence $\mathcal{H}$ représentant un domaine d'application et un objet `x` à expliquer (relativement à $\mathcal{H}$), il faut trouver une classe `C`, apparier `x` et `C` (dans la mesure du possible), puis expliquer le succès ou l'échec de l'appariement (le processus de classification, tel qu'il est présenté ici, correspond au processus d'*identification* de l'analyse de données, où il s'agit de découvrir la classe d'appartenance d'un individu [Leb 91]). La recherche de la « meilleure » classe est d'autant plus facile que l'ensemble des classes est ordonné : il existe donc une relation directe entre remémoration, catégorisation et classification.

Le processus de classification opère sur une hiérarchie $\mathcal{H} = (\mathcal{X}, \preceq, \omega)$ et cherche à mettre en évidence les dépendances implicites qui existent entre les objets dans $\mathcal{H}$, dépendances classes – classes et dépendances classes – instances.

Le processus de classification, qui permet de placer l'objet `x` dans la hiérarchie $\mathcal{H}$, se schématise comme suit :

$(\mathcal{X}, \preceq, \omega) \times \{\mathtt{x}\} \longrightarrow (\mathcal{X} \cup \{\mathtt{x}\}, \preceq, \omega)$.

Le processus de classification présente un aspect *constructif*, car il permet d'insérer un nouvel objet dans une hiérarchie d'objets $\mathcal{H}$, en accord avec les informations

présentes dans $\mathcal{H}$. Par conséquent, il peut servir de guide lors de la construction (manuelle) d'une hiérarchie, à l'image d'un processus de catégorisation hiérarchique incrémentale, et donner lieu ainsi à une construction *semi-automatique* de la hiérarchie.

Examinons maintenant l'influence du caractère nécessaire et/ou suffisant des propriétés attachées à une classe. Soit une classe `C` et `x` une instance de `C`. Dans ce cas, si `x` $\in$ `Extension(C)`, alors `x` *toutes* vérifie les propriétés de `C`, qui peuvent se voir comme des conditions *nécessaires* de l'appartenance de `x` à `Extension(C)`. Réciproquement, s'il s'avère que l'objet `y` vérifie *toutes* les propriétés de `C`, alors il est (éventuellement) possible de déduire que `y` $\in$ `Extension(C)` : les propriétés de `C` peuvent alors être vues comme *(collectivement) suffisantes* pour l'appartenance de `y` à l'extension de `C`.

Ainsi, si les propriétés d'une classe agissent comme des conditions nécessaires et suffisantes, alors le processus de classification est *envisageable*. Il faut noter ici qu'une telle hypothèse n'est pas toujours vérifiée : les propriétés des classes dans les langages de classes (variables d'instance par exemple) agissent comme des conditions nécessaires mais en aucun cas comme des conditions suffisantes [Euz 93].

Pour illustrer ce qui précède, considérons la classe `Vert-Blanc-Noir` et ses sous-classes `Vert`, `Blanc` et `Noir`.

`(i)` Si les propriétés des classes sont *nécessaires*, alors `Vert` ne recouvre que des objets verts : `x` $\in$ `Extension(Vert)` $\Longrightarrow$ `x est vert`. Il peut éventuellement y avoir des objets verts qui sont dans l'extension de la classe `Vert-Blanc-Noir`.

`(ii)` Si les propriétés des classes sont *nécessaires* et *suffisantes*, alors `Vert` recouvre *tous* les objets verts : `x` $\in$ `Extension(Vert)` $\Longleftrightarrow$ `x est vert`.

En particulier, si `x` $\in$ `Extension(Vert-Blanc-Noir)` et si `x est vert`, alors forcément `x` $\in$ `Extension(Vert)`.

2.2.5 Points de vue dans une hiérarchie

Si la relation organisant une hiérarchie est un ordre partiel, elle est antisymétrique, et dans ce cas `x R y, y R x` $\Longrightarrow$ `x = y` : deux classes ayant la même extension sont forcément *identiques*. Si la relation n'est pas antisymétrique, alors il est possible que deux classes de même extension possède une description — intension — différente, ce qui donne naissance à deux points de vue différents sur la même extension.

Il est souvent utile, voire nécessaire, de définir des *points de vue multiples* pour décrire les différentes facettes d'un concept du monde réel. Deux possibilités principales s'offrent au concepteur d'une base de connaissances. La première consiste à

décrire le concept à l'aide d'un ensemble de classes équivalentes : deux classes sont *équivalentes* si et seulement si elles ont la même extension. Les classes équivalentes déterminent des points de vue *locaux* sur le concept étudié. Par exemple, les points de vue locaux « polygone à 3 côtés » et « polygone à 3 angles » décrivent des classes équivalentes représentant le concept de triangle (voir aussi par exemple les points de vue présentés dans [Fer 88] [Mar 93]).

Les points de vue locaux correspondent à une dimension *horizontale* de la hiérarchie. À la dimension horizontale correspond une dimension *verticale*, qui est en rapport avec l'organisation hiérarchique des classes : plusieurs ordres partiels peuvent engendrer plusieurs hiérarchies, qui sont alors dites *croisées* [Nap 92]. Dans ce cas, une catégorie est prise en compte selon plusieurs points de vue *globaux*, où chaque point de vue correspond à une hiérarchie et à l'ordre partiel associé. Ainsi, étant donné une famille d'ordres partiels $\{\preceq_k\}$ (où $\mathtt{k} \in \mathtt{K}$, $\mathtt{K}$ étant un ensemble d'indices), et un ensemble de classes $\mathcal{X}$, les classes peuvent être organisées en une famille de hiérarchies croisées $\{\mathcal{H}_{\preceq_k} = (\mathcal{X}, \omega_{\preceq_k}, \preceq_k)\}$. Par exemple, les `choses` peuvent être divisées en `objets-vivants` et `objets-inanimés` ($\preceq_1$), en `objets-aériens`, `objets-terriens` et `objets-maritimes` ($\preceq_2$).

Les points de vue globaux servent à procéder à des regroupements mettant en lumière certaines propriétés. Ainsi, l'éléphant et la baleine, qui sont deux animaux relativement éloignés dans la classification des mammifères, se rapprochent pourtant pour (au moins) une propriété — assez inattendue, il est vrai — comme le montre l'extrait ci-dessous.

Bien que toute comparaison entre la baleine et l'éléphant soit absurde, dans le détail, l'éléphant se trouve vis-à-vis de la baleine dans la même position qu'un chien avec l'éléphant, on trouve néanmoins des similitudes curieuses entre eux, et parmi elles, celle-ci: le jet. C'est un fait connu que l'éléphant aspire de l'eau et la rejette souvent en poussière avec sa trompe. (Herman Melville, Moby Dick).

2.3 Le raisonnement hiérarchique

De façon générale, le *raisonnement hiérarchique* ou *raisonnement par classification* s'appréhende comme une procédure de déduction qui exploite les propriétés d'un univers où les connaissances sont représentées par des classes organisées en une hiérarchie [Att 86] [Att 91] [Hat 91]. Les hiérarchies d'héritage, les hiérarchies conceptuelles comme celles qui viennent d'être présentées, les hiérarchies conceptuelles des logiques terminologiques, ou encore les hiérarchies de composants dans la représentation d'objets composites sont des exemples de tels univers.

2.3.1 Opérations sur une hiérarchie

Les opérations principales qui sont à la base de la gestion d'une représentation hiérarchique sont les suivantes :

- *Subsomption* : déterminer qu'une classe `C1` est plus générale qu'une classe `C2` ; `C1` est alors le *subsumant* et `C2` le *subsumé*.
- *Classification* : chercher où se place dans la hiérarchie une nouvelle classe `C`.
- *Cohérence* et *instanciation* : une classe `C` n'a véritablement un sens que si elle peut avoir effectivement des instances. Réciproquement, il faut pouvoir déterminer les classes dont un objet `x` donné peut être instance.
- *Recherche d'information* : trouver les propriétés détenues par une classe, les restrictions associées à ces propriétés et/ou leurs valeurs.

Toutes les questions précédentes se ramènent fondamentalement à établir des relations de subsomption. La classification d'une classe se traduit par la recherche des subsumants les plus spécifiques et celles des subsumés les plus généraux de la classe. Une classe incohérente, comme celle qui dénote les objets physiques à la fois ronds et carrés par exemple, est subsumée par une classe spéciale, la classe notée $\bot$, qui ne possède aucune instance. Les propriétés qu'une classe `C` ne possède pas en propre sont détenues par les classes qui subsument `C`. D'où l'importance de la relation de subsomption dans une représentation hiérarchique.

Sur un plan plus général, une représentation hiérarchique peut se voir comme un *système logique*, qui fournit un processus de construction de termes — aspect *syntaxique* — et une procédure d'inférences, en l'occurrence la subsomption. Au processus de construction des termes peut être couplée une *fonction d'interprétation* qui donne la *sémantique* attachée à un terme ; cet aspect sémantique est donné par une fonction d'*extension* dans le cadre des logiques terminologiques [Neb 90a]. Comme dans le cas d'un système logique, se pose la question fondamentale de l'adéquation entre syntaxe et sémantique, qui se traduit par les propriétés de *correction* et *complétude* : le système est correct si les inférences effectuées sur les termes sont en accord avec la sémantique attachée aux termes ; le système est complet si toutes les inférences possibles sont découvertes par la procédure d'inférence. De nombreux travaux sur ces la vérification de ces deux propriétés ont été réalisés dans le cadre des logiques terminologiques : voir par exemple [Neb 90b] [Don 91a, Don 91b] et [Buc 93]. Ces problèmes sont devenus également une préoccupation dans le cadre des représentations hiérarchiques [Duc 95] [Euz 93, Euz 94].

2.3.2 La relation de subsomption

La *subsomption* est une relation générale qui permet d'organiser les classes en hiérarchies. Intuitivement, une classe `C` subsume une classe `D` si et seulement si l'extension de `C` contient nécessairement l'extension de `D`. Par exemple, la classe `Mammifère` subsume les classes `Chat` ou `Chien`, qui dénotent des familles de mammifères particuliers, l'un et l'autre étant des mammifères avant tout.

La relation de subsomption telle qu'elle est définie et utilisée dans les logiques terminologiques est détaillée dans [Neb 90a] (voir aussi l'étude présentée dans [Woo 91]). Par ailleurs, dans le cadre des représentations à objets, il est également possible de définir des relations de subsomption en considérant la structure (l'intension) des classes. Toutefois, nous ne nous étendrons pas plus avant sur ce sujet (voir par exemple [Nap 94b] et [Nap 95]).

2.3.3 Le cycle de classification

Le cycle de classification consiste à établir la position d'un objet `X`, classe ou instance, dans une hiérarchie $\mathcal{H}$. La définition de `x` induit sa position dans la hiérarchie, qui dépend de l'ensemble des *subsumants les plus spécifiques* ou SPS de `X` : une classe `A` fait partie des subsumants les plus spécifiques d'une classe `X` si `A` subsume `X` et si aucun subsumant de `X`, non équivalent à `A`, n'est subsumé par `A`.

La mise en œuvre du raisonnement par classification repose sur un *cycle* comprenant trois étapes :

`(i)` *instanciation* : création d'un nouvel objet `x` (classe ou instance),

`(ii)` *classification* : parcours de la hiérarchie $\mathcal{H}$ et mise en place de `x` dans $\mathcal{H}$, en accord avec l'ordre partiel $\preceq$,

`(iii)` *exploitation* : la mise en place de `x` déclenche des opérations de mise à jour d'objets interdépendants et/ou la production de nouveaux objets, ce qui ramène le cycle à sa première étape. Le cycle continue tant que les informations disponibles permettent de produire de nouveaux objets à classifier et que le but associé au problème à résoudre n'est pas atteint.

L'opération de classification se décompose en trois étapes principales, la recherche des subsumants les plus spécifiques de `x`, la recherche des subsumés les plus généraux de `x` (cette recherche est inutile si `x` est une instance), et la mise en place des nouvelles relations entre l'objet à classer `x`, ses subsumants et ses subsumés. Des algorithmes de classification sont détaillés dans [Lip 82], [Mac 88], [Baa 92] qui est repris dans [Baa 94], et enfin dans [Nap 94a].

Plus précisément, le parcours de la hiérarchie $\mathcal{H}$ s'effectue en profondeur, sur l'ensemble des classes de $\mathcal{H}$, en partant de la racine. Si la classe courante `C*` ne subsume pas `X`, alors la sous-hiérarchie de racine `C*`, notée $\mathcal{H}_{\texttt{C*}}$, est élaguée. Si `C*` subsume `X`, alors `C*` devient temporairement le subsumant le plus spécifique de `X` dans $\mathcal{H}_{\texttt{C*}}$, et le devient définitivement si aucun subsumant plus spécifique n'est trouvé dans $\mathcal{H}_{\texttt{C*}}$. Les tests de comparaison entre `X` et les classes présentes dans $\mathcal{H}$ portent sur la structure des objets comparés (voir par exemple [Neb 90a] ou [Nap 95]).

L'obtention des subsumants les plus spécifiques permet de focaliser la recherche des subsumés les plus généraux de `X`. Il suffit de ne considérer que l'ensemble des descendants $\mathcal{D}$ des SPS qui possèdent les mêmes propriétés que `X`. La mise en valeur des subsumés les plus généraux se fait comme suit : si `X` subsume un descendant `D` dans $\mathcal{D}$, alors `D` est un SPG et sa descendance est ignorée, sinon les descendants de `D` sont testés à leur tour, jusqu'à ce qu'un SPG soit trouvé ou bien jusqu'à ce qu'il n'y ait plus de descendant à tester ; le processus s'arrête lorsque toutes les classes de $\mathcal{D}$ ont été testées. Lorsque les SPS et les SPG de `X` ont été découverts, de nouveaux liens sont mis en place entre `X`, ses SPS et ses SPG. Si `X` est équivalent à une classe `C` déjà présente dans $\mathcal{H}_{\texttt{C}}$ — `C` fait alors à la fois partie des SPS et des SPG de `X` — `X` et `C` sont alors identifiés.

Par extension, il est envisageable d'appliquer le cycle de classification sur $\mathcal{H}$ avec plusieurs relations de subsomption $\preceq_{\mathcal{D}_k}$, correspondant à autant de points de vue globaux $\mathcal{D}_k$. Une telle application donne naissance à un ensemble de hiérarchies $\{\mathcal{H}_{\mathcal{D}_k} = (\mathcal{X}, \preceq_{\mathcal{D}_k}, \omega_{\mathcal{D}_k})\}$, où $\omega_{\mathcal{D}_k}$ correspond à l'élément maximal pour l'ordre partiel $\preceq_{\mathcal{D}_k}$, cet élément pouvant soit être arbitrairement fixé, soit être identique à la racine ω. Une hiérarchie $\mathcal{H}_{\mathcal{D}_k} = (\mathcal{X}, \preceq_{\mathcal{D}_k}, \omega_{\mathcal{D}_k})$ constitue une *dimension* dans l'ensemble des classes $\mathcal{X}$ reflétant le point de vue (global) $\mathcal{D}_k$. Le processus de construction de dimensions, qui s'appuie sur le cycle de classification, peut alors se voir comme une *classification multi-dimensionnelle*.

2.4 Bibliographie commentée

Outre les références déjà citées dans le texte, nous allons donner quelques indications bibliographiques qui concernent essentiellement la catégorisation et la classification sous les différentes formes abordées ici. Ainsi, parmi les livres de référence en analyse de données, citons les ancêtres [Ler 70] [Jar 71] et [Ben 73], les (désormais) classiques [Did 82] [Jam 89] [Sap 90], et le récent (et très bien fait) [Leb 95]. À cela, il faut ajouter les articles de synthèse [Bar 84] [Gor 87], les livres [Mur 85] [Bar 88], et les deux articles [Lec 85] [Bar 86] portant sur la comparaison de hiérarchies.

Les structures ordonnées et les graphes forment des structures mathématiques de base dans l'étude de la classification, qui sont détaillées dans [Bar 70] [Dav 90] [Gon 95].

Côté intelligence artificielle, un des premiers livres où figure la classification comme mode de raisonnement à part entière est [Hat 91] (eh oui !). Il est également question de classification (heuristique) dans [Cla 85], de rapports entre raisonnement par classification et raisonnement logique dans [Gom 91]. La classification est examinée en détail dans le très récent et très imposant livre sur les systèmes à base de connaissances [Ste 95] ; classification et catégorisation sont également discutées de façon approfondie dans les livres dédiés à l'apprentissage, dont les classiques [Mic 84] [Mic 86], et le récent [Lan 96], qui, en outre, est un très bon livre sur l'ensemble des techniques d'apprentissage. Côté logiques terminologiques, la référence de base reste [Neb 90a], mais d'autres détails et préoccupations peuvent être trouvés (entre autres) dans [Bra 91] [Woo 92].

La construction de hiérarchies présente également des rapports avec la théorie des treillis de Galois comme en témoignent les articles suivant [Wil 92] [Dic 94] [God 95].

Les sciences de la vie ont de tout temps été à l'origine d'études de tous ordres sur la classification — la taxinomie — comme le montre d'ailleurs fort bien le livre [Ben 73]. Comme il est bien sûr difficile de faire un choix parmi l'abondante littérature sur le sujet, nous nous contenterons de citer les livres et papiers qui sont à notre proximité en ce moment : [Bar 91] [Dar 93] et [Car 95] sur les arbres (phylogénétiques), [Jan 80], [Mat 87] et [dG 90] sur la systématique zoologique et le cladisme, et [May 95] pour une vue encyclopédique — et bon marché — sur le sujet.

Pour finir, citons quelques papiers qui essaient d'établir des liens entre classification symbolique, classification numérique et classification neuromimétique : [Cha 88] [Fis 89] [Moo 89] [Wei 89].

Bibliographie

[Att 86] G. Attardi, A. Corradini, S. Diomedi, et M. Simi. Taxonomic Reasoning. *Proceedings of the 7th European Conference on Artificial Intelligence (ECAI'86), Brighton, England*, pages 236–245, 1986.

[Att 91] G. Attardi. An Analysis of Taxonomic Reasoning. M. Lenzerini, D. Nardi, et M. Simi, éditeurs, *Inheritance Hierarchies in Knowledge Representation and Programming Languages*, pages 29–49. John Wiley & Sons Ltd, Chichester, West Sussex, 1991.

[Baa 92] F. Baader, B. Hollunder, B. Nebel, H.-J. Profitlich, et E. Franconi. An Empirical Analysis of Optimization Techniques for Terminological Representation Systems. *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning (KR'92), Cambridge, Massachusetts*, pages 270–281, 1992.

[Baa 94] F. Baader, B. Hollunder, B. Nebel, H.-J. Profitlich, et E. Franconi. An Empirical Analysis of Optimization Techniques for Terminological Representation Systems. *Applied Intelligence*, 4(2):109–132, 1994.

[Bar 70] M. Barbut et B. Monjardet. *Ordre et classification – Algèbre et combinatoire (2 tomes).* Hachette, Paris, 1970.

[Bar 84] J.P. Barthélemy, B. Leclerc, et B. Monjardet. Ensembles ordonnés et taxonomie mathématique. *Annals of Discrete Mathematics*, 23:523–548, 1984.

[Bar 86] J.P. Barthélemy, B. Leclerc, et B. Monjardet. On the Use of Ordered Sets in Problems of Comparison and Consensus of Classifications. *Journal of Classification*, 3:187–224, 1986.

[Bar 88] J.P. Barthélemy et A. Guénoche. *Les arbres et les représentations des proximités*. Masson, Paris, 1988.

[Bar 91] J.-P. Barthélemy. Similitudes, arbres et typicalité. D. Dubois, éditeur, *Sémantique et cognition – Catégories, prototypes, typicalité*, pages 205–224. Éditions du CNRS, Paris, 1991.

[Ben 73] J.-P. Benzecri. *L'analyse des données – La taxinomie*. Dunod, Paris, 1973.

[Bra 91] R.J. Brachman, D.L. McGuinness, P.F. Patel-Schneider, L.A. Resnick, et A. Borgida. Living with CLASSIC: When and How to Use a KL-ONE Language. J. Sowa, éditeur, *Principles of Semantic Networks: Explorations in the Representation of Knowledge*, pages 401–456. Morgan Kaufmann Publishers, Inc., San Mateo, California, 1991.

[Buc 93] M. Buchheit, F.M. Donini, et A. Schaerf. Decidable Reasoning in Terminological Knowledge Representation Systems. *Journal of Artificial Intelligence Research*, 1:109–138, 1993.

[Car 95] G. Caraux, O. Gascuel, G. Andrieu, et D. Levy. Approche informatiques de la reconstruction d'arbres phylogénétiques à partir de données moléculaires. *Techniques et Sciences Informatiques*, 14(2):113–139, 1995.

[Cha 88] B. Chandrasekaran et A. Goel. From Numbers to Symbols to Knowledge Structures: Artificial Intelligence Perspectives on the Classification Task. *IEEE Transactions on Systems, Man and Cybernetics*, 18(3):415–424, 1988.

[Cla 85] W.J. Clancey. Heuristic Classification. *Artificial Intelligence*, 27:289–350, 1985.

[Dar 93] P. Darlu et P. Tassy. *Reconstruction phylogénétique : concepts et méthodes*. Masson, Paris, 1993.

[Dav 90] B.A. Davey et H.A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, Cambridge, UK, 1990.

[dG 90] M. d'Ukedem Gevers. L'analyse cladistique – Problèmes et solutions heuristiques informatisées. *Biosystema (Société Française de Systématique)*, 4, 1990.

[Dic 94] H. Dicky, C. Dony, M. Huchard, et T. Libourel. ARES, un alogorithme d'Ajout avec REStructuration dans les hiérarchies de classes. *Actes de la conférence Langages et Modèles à Objets (LMO'94), Grenoble*, pages 125–136, 1994.

[Did 82] E. Diday, J. Lemaire, J. Pouget, et F. Testu. *Eléments d'analyse des données*. Dunod, Paris, 1982.

[Don 91a] F.M. Donini, M. Lenzerini, D. Nardi, et W. Nutt. The Complexity of Concept Languages. *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning (KR'91), Cambridge, Massachusetts*, pages 151–162, 1991.

[Don 91b] F.M. Donini, M. Lenzerini, D. Nardi, et W. Nutt. Tractable Concept Languages. *Proceedings of the 12th IJCAI, Sydney, Australia*, pages 458–463, 1991.

[Dub 91] D. Dubois, éditeur. *Sémantique et cognition – Catégories, prototypes, typicalité*. Editions du CNRS, Paris, 1991.

[Duc 89] R. Ducournau et M. Habib. La multiplicité de l'héritage dans les langages à objets. *Techniques et Sciences Informatiques*, 8(1):41–62, 1989.

[Duc 95] R. Ducournau. Les systèmes classificatoires. Rapport de Recherche no. 95-038, LIRM, Montpellier, 1995.

[Euz 93] J. Euzenat. Définition abstraite de la classification et son application aux taxonomies d'objets. *Actes de la conférence Représentations Par Objets (RPO'93), La Grande Motte*, pages 235–246, 1993.

[Euz 94] J. Euzenat. Classification dans les représentations par objets : produits de systèmes classificatoires. *Actes du 9ème Congrès AFCET Reconnaissance des Formes et Intelligence Artificielle (RFIA'94), Paris*, pages 185–196, 1994.

[Fer 88] J. Ferber et P. Volle. Using Coreference in Object Oriented Representations. *Proceedings of the 8th European Conference on Artificial Intelligence (ECAI'88), Munich, Germany*, pages 238–240, 1988.

[Fis 89] D.H. Fisher et K.B. McKusick. An Empirical Comparison of ID3 and Back-propagation. *Proceedings of the 11th IJCAI, Detroit, Michigan*, pages 788–793, 1989.

[God 95] R. Godin, G. Mineau, R. Missaoui, et H. Mili. Méthodes de classification conceptuelle basées sur les treillis de Galois et applications. *Revue d'intelligence artificielle*, 9(2):105–137, 1995.

[Gom 91] F. Gomez et C. Segami. Classification-Based Reasoning. *IEEE Transactions on Systems, Man and Cybernetics*, 21(3):644–659, 1991.

[Gon 95] M. Gondran et M. Minoux. *Graphes et algorithmes (3ième édition revue et augmentée)*. Éditions Eyrolles, Paris, 1995.

[Gor 87] A.D. Gordon. A Review of Hierarchical Classification. *Journal of the Royal Statistical Society (A)*, 150(2):119–137, 1987.

[Hat 91] J.-P. Haton, N. Bouzid, F. Charpillet, M.-C. Haton, B. Lâasri, H. Lâasri, P. Marquis, T. Mondot, et A. Napoli. *Le raisonnement en intelligence artificielle*. InterEditions, Paris, 1991.

[Jam 89] M. Jambu. *Exploration informatique et statistiques des données*. Dunod, Paris, 1989.

[Jan 80] P. Janvier, P. Tassy, et H. Thomas. Le cladisme. *La Recherche*, 117:1396–1406, Décembre 1980.

[Jar 71] N. Jardine et R. Sibson. *Mathematical Taxonomy*. John Wiley & Sons Ltd, London, 1971.

[Kle 90] G. Kleiber. *La sémantique du prototype*. Presses Universitaires de France, Paris, 1990.

[Lan 96] P. Langley. *Elements of Machine Learning* . Morgan Kaufmann Publishers, San Francisco, California, 1996.

[Leb 91] J. Lebbe. Représentation des concepts en biologie et médecine (Introduction à l'analyse des connaissances et à l'identification assistée par ordinateur). Thèse de l'Université Pierre et Marie Curie (Paris 6), 1991.

[Leb 95] L. Lebart, A. Morineau, et M. Pinon. *Statistique exploratoire multidimensionnelle*. Dunod, Paris, 1995.

[Lec 85] B. Leclerc. La comparaison des hiérarchies : indices et métriques. *Mathématiques et Sciences Humaines*, 92:5–40, 1985.

[Ler 70] I.C. Lerman. *Les bases de la classification automatique.* Gauthier-Vilars, Paris, 1970.

[Lip 82] T. Lipkis. A KL-ONE Classifier. J.G. Schmolze et R.J. Brachman, éditeurs, *Proceedings of the 1981 KL-ONE Workshop, Jackson, New Hampshire*, pages 126–143, 1982.

[Mac 88] R. MacGregor. A Deductive Pattern Matcher. *Proceedings of AAAI'88, St Paul, Minnesota*, pages 403–408, 1988.

[Mar 93] O. Mariño. Raisonnement classificatoire dans une représentation à objets multi-points de vue. Thèse de Doctorat, Université Joseph Fourier, Grenoble, 1993.

[Mas 89] G. Masini, A. Napoli, D. Colnet, D. Léonard, et K. Tombre. *Les langages à objets.* InterEditions, Paris, 1989.

[Mat 87] L. Matile, P. Tassy, et D. Goujet. Introduction à la systématique zoologique (Concepts, principes, méthodes). *Biosystema (Société Française de Systématique)*, 1, 1987.

[May 95] E. Mayr. *Histoire de la biologie (Diversité, évolution et hérédité).* Le livre de Poche – Fayard, Paris, 1995.

[Mec 93] I. Van Mechelen, J. Hampton, R.S. Michalski, et P. Theuns, éditeurs. *Categories and Concepts. Theoretical Views and Inductive Data Analysis.* Academic Press, London, 1993.

[Mic 84] R.S. Michalski, J.G. Carbonell, et T.M. Mitchell, éditeurs. *Machine Learning, an Artificial Intelligence Approach, Volume I.* Springer-Verlag, Berlin, 1984.

[Mic 86] R.S. Michalski, J.G. Carbonell, et T.M. Mitchell, éditeurs. *Machine Learning, an Artificial Intelligence Approach, Volume II.* Morgan Kaufmann Publishers, Inc., Los Altos, California, 1986.

[Moo 89] R. Mooney, J. Shavlik, G. Towell, et A. Gove. An Experimental Comparison of Symbolic and Connectionist Learning Algorithms. *Proceedings of the 11th IJCAI, Detroit, Michigan*, pages 775–780, 1989.

[Mur 85] F. Murtagh. *Multidimensional Clustering Algorithms.* Compstat Lectures 4 (Lectures in Computational Statistics), Edited by J.M. Chambers, J. Gordesch, A. Klas, L. Lebart and P.P. Sint. Physica–Verlag, Wuerzburg, 1985.

[Nap 92] A. Napoli. *Représentations à objets et raisonnement par classification en intelligence artificielle.* Thèse de Doctorat d'État, Université de Nancy 1, 1992.

[Nap 93] A. Napoli et C. Laurenço. Représentations à objets et classification – Conception d'un système d'aide à la planification de synthèses organiques. *Revue d'intelligence artificielle*, 7(2):175–221, 1993.

[Nap 94a] A. Napoli. Catégorisation, raisonnement par classification et raisonnement à partir de cas. *Journées Acquisition – Validation – Apprentissage (JAVA'94), Strasbourg*, pages E1–E14, 1994.

[Nap 94b] A. Napoli, C. Laurenço, et R. Ducournau. An object-based representation system for organic synthesis planning. *International Journal of Human-Computer Studies*, 41(1/2):5–32, 1994.

[Nap 95] A. Napoli. Étude formelle des rapports entre raisonnement par classification et raisonnement à partir de cas : les problèmes d'organisation de la mémoire. I. Bichindaritz, éditeur, *Actes du quatrième séminaire français sur le raisonnement à partir de cas*, pages 17–24. Université de Paris 5 – AFCET, 1995.

[Neb 90a] B. Nebel. *Reasoning and Revision in Hybrid Representation Systems.* Lecture Notes in Computer Science 422. Springer-Verlag, Berlin, 1990.

[Neb 90b] B. Nebel. Terminological Reasoning is Inherently Intractable. *Artificial Intelligence*, 43(2):235–249, 1990.

[Sap 90] G. Saporta. *Probalités, analyse des données et statistique.* Technip, Paris, 1990.

[Ste 95] M. Stefik. *Introduction to Knowledge Systems.* Morgan Kaufmann Publishers, Inc., San Francisco, California, 1995.

[Vig 91] R. Vignes. Caractérisation automatique de groupes biologiques. Thèse de l'Université Pierre et Marie Curie (Paris 6), 1991.

[Wei 89] S.M. Weiss et I. Kapouleas. An Empirical Comparison of Pattern Recognition, Neural Nets, and Machine Learning Classification Methods. *Proceedings of the 11th IJCAI, Detroit, Michigan*, pages 781–787, 1989.

[Wil 92] R. Wille. Concept Lattices and Conceptual Knowledge Systems. *Computers & Mathematics With Applications*, 23(6–9):493–515, 1992.

[Woo 91] W.A. Woods. Understanding Subsumption and Taxonomy: A Framework for Progress. J. Sowa, éditeur, *Principles of Semantic Networks: Explorations in the Representation of Knowledge*, pages 45–94. Morgan Kaufmann Publishers, Inc., San Mateo, California, 1991.

[Woo 92] W.A. Woods et J.G Schmolze. The KL-ONE Family. *Computers & Mathematics With Applications*, 23(2–5):133–177, 1992.

Réseaux neuromimétiques et classification

Frédéric Alexandre
CRIN-CNRS & INRIA-Lorraine
C.-él. : falex@loria.fr

3.1 Introduction

Les réseaux de neurones artificiels sont couramment utilisés pour des tâches de classification. Jean-François Remm et Jean-Daniel Kant en donneront deux exemples précis aux chapitres 3 et 6. Le but de ce chapitre est d'introduire le formalisme connexionniste, ce qui permettra ensuite de mieux comprendre les chapitres 3 et 6, directement orientés vers leur but de relations entre classification et connexionnisme.

Je donnerai ensuite un aperçu des premiers modèles élémentaires dont la relation avec la classification est directe. Cet aperçu sera tout d'abord lié aux règles de fonctionnement du neurone formel qui peut réaliser un produit scalaire ou une fonction distance et qui peut donc, dans le cas du perceptron, faire une séparation linéaire de classes et dans le cas des cartes de Kohonen, faire du *clustering*.

Je donnerai enfin un aperçu d'autres modèles plus avancés, comme les modèles bilinéaires, les mémoires associatives et les fonctions à base radiale.

3.2 Formalisme élémentaire

Le connexionnisme peut être défini comme le calcul distribué d'unités simples, regroupées en réseau. Par analogie avec la biologie, ces unités sont appelées neurones formels mais c'est généralement la seule analogie qui existe.

Un neurone formel est caractérisé par :

- la nature de ses entrées et ses sorties ;
- sa fonction d'entrée ;
- sa fonction de sortie.

3.2.1 Evaluation des entrées

Ce calcul est effectué sur les entrées $(e_i)_{i=1,n}$, qui peuvent être binaires ou réelles. E est la fonction d'entrée qui définit l'activation interne du neurone. E peut être une fonction booléenne des entrées (e_i). Sinon, on utilise des paramètres de pondération W_i permettant à l'entrée totale d'être une fonction linéaire E=$\Sigma W_i e_i$ ou affine E=$\Sigma W_i e_i - a$. Ces sommes pondérées correspondent en fait à un produit scalaire entre le vecteur d'entrée et le vecteur poids (W). Ceci est principalement utilisé dans les réseaux de type perceptron que l'on verra plus loin. Une autre combinaison possible, utilisée dans les cartes de Kohonen, correspond au calcul d'une distance entre ces deux vecteurs $(E)^2$=$\Sigma(W_i - e_i)^2$. On remarquera que, si les poids sont normalisés, chercher la distance minimum pour plusieurs vecteurs poids différents (ce qu'on fait dans les cartes de Kohonen) revient à chercher le produit scalaire maximum (comme on le fait avec le perceptron). En effet, $(W - e)^2 = (W)^2 + (e)^2 - 2eW$.

Ces fonctions d'entrée (distance et produit scalaire) sont les plus fréquemment rencontrées. On trouve cependant d'autres fonctions d'entrée, comme dans le cas des réseaux dits RBF (radial basis function) où l'entrée est convoluée avec une fonction à base radiale, une gaussienne par exemple.

3.2.2 Evaluation de la sortie

La fonction d'activation f définit l'état du neurone et donc sa sortie, en fonction de son entrée totale E. On la représente généralement par une fonction impaire et croissante pouvant être:

- une fonction binaire à seuil
- une fonction linéaire à seuil(s) comprenant des domaines linéaires et des domaines seuillés
- une fonction stochastique:

  f(x)=1 avec la probabilité $1/(1 + e^{-x/T})$

  f(x)=0 sinon.
- une fonction sigmoïde:

  $f(x) = a(e^{kx} - 1)/(e^{kx} + 1)$

On a donc, globalement, la sortie en fonction des entrées:
$Y_j = f(E_j) = f(\Sigma W_{ij} e_i)$

Les règles d'apprentissage

Il convient tout d'abord de faire la distinction entre apprentissage supervisé et non supervisé. Dans le premier cas, l'apprentissage s'effectue sous le contrôle d'un professeur qui fixe le comportement désiré du réseau. Ceci donne donc une erreur (différence entre le comportement réel et désiré) qui sera à la base de l'apprentissage (sous la forme d'une fonction de coût ou d'un signal d'erreur). Ceci donnera donc des fonctions de classification supervisées.

Dans le deuxième cas, le réseau extrait de manière autonome l'information pertinente du flux d'information lui parvenant. Il va donc extraire les régularités de ces flux, ce qui va donner lui à des fonctions proches du clustering.

Dans tous les réseaux, l'apprentissage s'effectue en modifiant les poids W_i des connexions entre les neurones et la mémorisation dépend, à un instant donné, de la force de ces liens. Cette modification est réalisée selon deux principes d'inspiration différente.

1. Inspiration biologique :

   Il s'agit de la règle de Hebb qui a proposé, il y a une quarantaine d'années, le mécanisme de mémorisation suivant : un neurone ne mémorise une association que si les signaux émis par une source présynaptique se produisent en même temps qu'une décharge dans l'élément postsynaptique. Dans un réseau de neurones formels, cette règle est appliquée en renforçant la connexion de deux neurones activés au même moment.

   On a donc une variation du poids de la forme :

   $W_{ij}(t+1) = W_{ij}(t) + \alpha E_i E_j$

   où α représente le pas d'incrémentation du poids.

2. Inspiration mathématique :

   Il s'agit alors de considérer que le réseau effectue une fonction de transfert sur les entrées pour calculer les sorties. Les paramètres de cette fonction, c'est-à-dire les poids des connexions, sont modifiés en terme de minimisation de fonctions de coûts, définies entre la sortie réelle et la sortie désirée du réseau. Il s'agit d'un apprentissage supervisé.

Ces deux lois d'apprentissage ne sont pas essentielles ici pour comprendre les liens entre connexionnisme et classification. Il suffit seulement de comprendre que des algorithmes d'apprentissage permettent de déterminer automatiquement les paramètres

libres du réseau (les poids) qui correspondent en fait à des paramètres permettant de définir les frontières des classes.

Sur la base de ce formalisme élémentaire, nous pouvons maintenant présenter quelques modèles connexionnistes classiques et montrer leurs liens avec la classification.

Classiquement, les réseaux sont utilisés comme mémoires associatives. Leur rôle peut être de reconstruire une information bruitée (auto-association) ou d'associer deux types d'informations différents (hétéro-association). Ceci permet de déboucher sur des applications de reconnaissance des formes, de traitement du signal, de classification, de traitement d'images... Un autre domaine d'action des réseaux est la satisfaction de contraintes avec des applications à l'optimisation, l'aide à la décision, la prévision...

3.3 Les réseaux à couches

Une structuration en couches tend à rendre compte des différents traitements que l'on peut effectuer en cascade sur un ensemble d'informations. Ces informations sont proposées sur une couche extrême, appelée couche d'entrée; elles sont ensuite traitées par un nombre variable de couches intermédiaires ou couches cachées. Le résultat apparaît sur l'autre couche extrême, la couche de sortie.

Généralement, ces réseaux sont utilisés pour des tâches d'association (hétéro-association dans la majorité des cas) et sont, pour cette raison, souvent appelés réseaux associatifs. Etudions en quelques uns.

3.3.1 Le perceptron

Historiquement, le perceptron est le premier réseau efficace a avoir été proposé et étudié en détail.

Il comprend une couche d'entrée et une couche de sortie. Les connexions entre ces deux couches sont modifiables et bidirectionnelles. Les unités ont des sorties binaires. Les unités de la couche de sortie réalisent une fonction à seuil. C'est la couche de sortie qui remplit la tâche de classification. Les unités sont évaluées parallèlement. Les connexions sont modifiées par un apprentissage supervisé selon le principe de correction d'erreurs : si une unité de la couche de sortie n'est pas dans l'état désiré, toutes les liaisons que cette unité réalise avec les unités de la couche d'entrée sont augmentées ou diminuées selon le type d'action que l'unité de la couche d'entrée réalisait sur l'unité de sortie considérée, dans un sens permettant de favoriser une

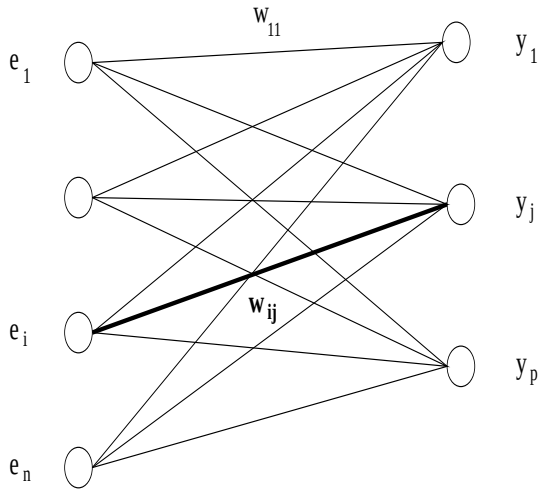


FIG. 3.4 – *Perceptron monocouche*

réponse exacte. Analytiquement, cette modification potentielle de poids synaptique peut être décrite par :

$$W_i(t+1) = W_i(t) + k(d-s)e_i$$

avec d : sortie désirée, s : sortie réelle, et k : réel positif représentant le pas de modification des poids.

De par son rôle d'unité linéaire à seuil, chaque unité de la couche de sortie réalise une partition de l'espace de ses entrées en deux classes séparées par l'hyperplan d'équation

$$\Sigma_i W_i e_i - seuil = 0$$

Ainsi, n unités de sortie permettent de séparer linéairement 2^n classes.

3.3.2 Les réseaux multicouches

L'unité choisie est du même genre que pour le perceptron. Seule différence, la fonction à seuil cède la place à une fonction sigmoïde de la forme :

$$f(x) = k(e^{kx} - 1)/(e^{kx} + 1)$$

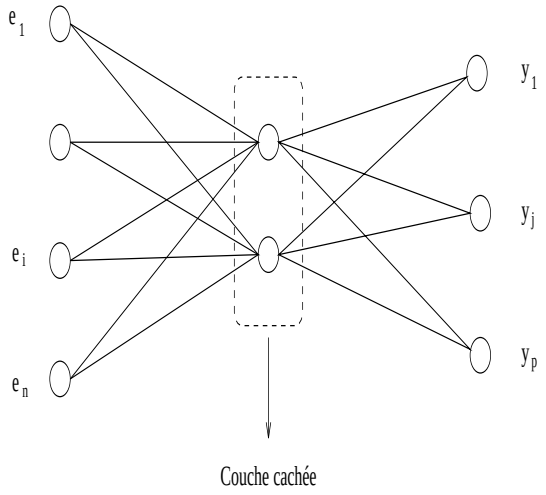


FIG. 3.5 – *Perceptron multicouche*

Or, les fonctions sigmoïdes sont dérivables...

L'apprentissage va consister, pour minimiser l'erreur commise, à réaliser une descente du gradient, c'est-à-dire à calculer la dérivée de l'erreur totale commise par rapport à un poids particulier. Un neurone possédant une fonction d'activation dérivable (comme les sigmoïdes) permet de réaliser simplement cette opération.

La mise au point de cet algorithme a entraîné une utilisation intensive des réseaux multicouches dans des domaines très variés. Il n'en reste pas moins que cette technique possède des inconvénients et des limitations pas encore surmontés. Ainsi, il n'a pas encore été possible de déterminer automatiquement un certain nombre de paramètres fondamentaux pour un réseau multicouche.

Ces réseaux sont également principalement utilisés pour des tâches de mises en correspondance et en particulier de classification. Jean-François Remm montrera la relation entre ces réseaux et les classifieurs bayésiens.

3.3.3 Autres réseaux à couches

Les réseaux à fonction à base radiale (RBF) sont des réseaux à deux couches. La première couche est en ensemble de neurones réalisant une convolution avec une fonction gaussienne. Cette fonction est apprise de manière non supervisée d'une manière similaire à Kohonen (cf. plus bas). Ensuite, la deuxième couche réalise simplement

une combinaison linéaire des neurones de la couche cachée. Ici, la séparation des classes ne se fait donc plus par des hyperplans, mais par des hypersphères.

Nous décrivons enfin rapidement deux modèles. Le premier introduit la classification non supervisée. Le second sera plus amplement décrit par Jean-Daniel Kant.

3.4 Le modèle de Kohonen

Ce modèle correspond à une couche de neurones apprenant de manière non supervisée. Chaque neurone calcule une distance entre le stimulus courant et son vecteur poids $(E)^2{=}\Sigma(W_i - e_i)^2$. Le neurone ayant la distance minimum l'emporte. Il règle ses poids de manière à se rapprocher encore plus du stimulus courant $W_i(t+1) = W_i(t) + k(W_i - e_i)$. Ainsi, chaque famille de poids de chaque neurone (cette famille étant de même dimension qu'en entrée) se place au centre de gravité de l'ensemble d'entrées le plus proche de la valeur initiale de cette famille de poids. Ceci donne donc lieu à une opération de clustering. Dans sa version topologique, chaque neurone gagnant incite ses voisins à modifier également ses poids dans le même sens. Ceci permet donc à la couche entière de s'auto-organiser de manière topologique. Si cette couche est surdimensionnée par rapport à l'ensemble des clusters d'entrée, la couche de Kohonen réalise en fait une approximation de la distribution de probabilité de ces entrées.

A partir de cette version non supervisée, Kohonen propose ensuite des versions supervisées (LVQ pour Learned Vector Quantization). Ces versions consistent, avec plusieurs variantes, à améliorer et rendre plus robuste cette séparation en classes, à partir de la solution proposée par l'apprentissage non supervisé. Cette dernière est tout d'abord étiquetée (donc de manière supervisée) puis chaque exemple et repris et ceux qui se trouvent trop prêt de la frontière entraînent une modification de poids supervisée, recentrant cet exemple. On obtient ainsi un classifieur supervisé.

3.5 ART

L'Adaptive Resonance Theory permet d'introduire la notion de réseau prototypaux. Ici, il s'agit de construire de manière ad hoc des neurones possédant des masques sur les entrées, réglés de manière à ce que ces neurones soient des prototypes de certaines de ces entrées. La notion de classe est donc construite ici autour d'un représentant (correspondant au neurone prototype) et d'une distance maximum

à ce représentant permettant d'appartenir à cette classe (appelée seuil de vigilance). Le modèle ART existe sous forme binaire et continue.

L'algorithme d'apprentissage consiste donc à présenter un exemple, à chercher le neurone le plus proche (il s'agit donc, comme pour Kohonen, de neurones calculant des distances entre l'exemple présenté et son vecteur poids). Si ce vecteur est suffisamment proche (distance inférieure au seuil de vigilance), alors cet exemple est assimilé à la classe représentée par ce neurone, sinon un nouveau neurone est créé et son vecteur poids est initialisé de manière à ce que ce neurone devienne un prototype de cet exemple.

Cette technique correspond également à une technique classique de classification.

3.6 Conclusion

Nous avons donné ici quelques indices permettant d'entrevoir les relations existant entre connexionnisme et classification. Ceci sera poursuivi et approfondi avec le chapitre 5 consacré aux *probabilités et réseaux de neurones* et dans le chapitre 6 consacré à *l'Introduction au problème de l'interprétation des données en classification.*

On voit globalement se dessiner deux stratégies. Soit les neurones réalisent un produit scalaire entre leurs poids et leurs entrées et dans ce cas les poids permettent de définir les frontières des classes. Soit les neurones calculent une distance entre leurs poids et leurs entrées et dans ce cas, les poids représentent le centre de la classe.

On remarque aussi que, sur cette base, les différents algorithmes d'apprentissage permettent de manipuler ces frontières et ces centres de classe de manière assez analogue à des algorithmes classiques de classification. Cependant, une différence essentielle est que ces algorithmes classiques fonctionnent souvent avec des lois linéaires, alors que les neurones formels utilisent souvent des fonctions non linéaires, comme les sigmoïdes.

Enfin, il faut préciser que ces analogies ne s'arrètent pas au seul domaine de la classification. Plus généralement, c'est avec le domaine des statistiques et de l'analyse de données que le connexionnisme développe des liens d'analogie nombreux.

Bibliographie

[Hertz 91] J. Hertz, A. Krogh, R. Palmer. *Introduction to the theory of neural computation.* Addison Wesley, 1991.

[Kohonen 84] T. Kohonen. *Self Organization and Associative Memory.* Springer, 1984.

[Kohonen 88] T. Kohonen. *An introduction to neural computing.* Neural Networks, 1, p. 3-16, 1988.

[LeCun 87] Y. LeCun. *Modèles connexionnistes de l'apprentissage.* Thèse de Doctorat, Université Paris VI, 1987.

[Rosenblatt 62] F. Rosenblatt. *A comparison of several perceptron models.* Self Organizing Systems 1962, ed. Yovits, Jacobi, Goldstein, Spatan Books, p. 463-484.

[Rumelhart 86] D. Rumelhart, J. MacClelland. *Parallel distributed processing.* MIT Press, Cambridge, 1986.

Classification et apprentissage : la formation de concepts structurés

Alain Ketterlin
LSIIT, URA CNRS 1871
Université Louis Pasteur – Strasbourg I
7, rue René Descartes, 67084 Strasbourg Cedex
C.-él. : alain@dpt-info.u-strasbg.fr

4.1 Problématique

Notre exposé portera sur l'apprentissage non supervisé. Le problème est de construire une représentation conceptuelle dans un domaine, à partir de données recueillies dans ce domaine. Plus précisement, cette tâche est appelée *formation de concepts* si le résultat est une hiérarchie de classes (ou de concepts – les deux termes sont utilisés indifféremment ici), et que le processus est incrémental (c'est-à-dire que la hiérarchie est modifiée par chaque nouvelle observation, mais disponible à tout moment). L'objectif de tels algorithmes est la construction automatique de bases de connaissances, sur des bases empiriques.

4.2 Stratégies et formalismes

Cette formulation du probleme n'est pas sans rappeler ce domaine de l'analyse de données appelé *classification automatique*. De fait, les problématiques sont les mêmes. Mais les algorithmes d'apprentissage se préoccupent principalement de la forme des données et de la forme des connaissances induites : le but est dans ce cas de construire automatiquement des bases de connaissances, et le langage employé est donc de première importance. On pourrait, de façon reductrice peut-être, dire que les méthodes statistiques consistent à regrouper les observations, alors que les méthodes d'apprentissage consistent à découvrir et à décrire les regroupements pertinents. Dans cette section, nous allons brièvement tracer quelques parallèles entre les algorithmes employés dans les deux disciplines. Nous allons pour cela évoquer trois algorithmes

d'apprentissage. Dans chaque cas, nous allons tenter de rapprocher l'algorithme d'une méthode statistique, et de discerner ce qui distingue les données analysées et les classes construites des données et résultats statistiques.

4.2.1 Cluster

L'algorithme Cluster permet de construire, à partir de la donnée d'un certain nombre d'observations, un ensemble de règles définissant des classes. L'algorithme de base est très similaire aux procédures statistiques de type *nuées dynamiques*, c'est-à-dire qu'il procède par sélection de noyaux, auxquels il agrège les observations disponibles. Il calcule ensuite, à partir des ensembles affectés à chaque noyau, une nouvelle « position » de ce noyaux, et réitère le processus d'affectation des observations. On retrouve dans ce cas une stratégie bien connue des statisticiens.

Pour ce qui est de la représentation des données, Cluster utilise un langage que son auteur appelle APC (*Annotated Predicate Calculus*). Ce langage permet d'exprimer chaque donnée par une conjonction de termes, chaque étant l'affectation d'une valeur à un attribut. Ces valeurs peuvent être de type discret ou hiérarchisé. Le résultat de l'apprentissage est, comme nous l'avons mentionné, une règle définissant chaque classe découverte. Au cours de chaque cycle, Cluster construit pour chaque noyau, une règle permettant de distinguer ce noyau de tous les autres. Ce sont ces règles qui sont utilisées ensuite pour réaffecter les observations aux noyaux (par un test booléen simple). De plus, l'itération se pousuit jusqu'à ce qu'un critère de terminaison défini par l'utilisateur soit vérifié : une tel critère inclut typiquement des considérations propres à la complexité de la règle (c'est-à-dire le nombre de sélecteurs), mais permet également de pondérer l'importance des différents attributs. C'est par le biais d'un tel critère que Cluster est capable de *classification conceptuelle*.

4.2.2 KBG

L'algorithme KBG a été développé dans le but avoué d'automatiser la construction de bases de connaissances. En fait, il n'y a pas d'algorithme KBG à proprement parler, puisque ce système est fait pour pouvoir utiliser n'importe quelle stratégie statistique, à partir du moment où celle-ci est basée sur une notion de similarité. Dans sa description initiale, l'auteur a choisi une variante de la classification ascendante hiérarchique, mais insiste sur le fait que d'autres algorithmes feraient aussi bien l'affaire.

On aura donc compris que l'apport de KBG se situe essentiellement au niveau de la représentation des connaissances. Considérant que les algorithmes de classification statistique sont paramétrés par leur mesure de similarité, l'auteur de KBG a développé une mesure de ce type lorsque chaque objet est une conjonction de termes prédicatifs. Il y est donc possible de mesurer la similarité entre deux exemples, de comparer les différentes valeurs obtenues de cette façon, et de généraliser en un nouveau concept (c'est-à-dire une conjonction de prédicats variabilisés) les exemples les plus similaires.

4.2.3 Cobweb

L'algorithme Cobweb se distingue des deux précédents, puisqu'il met l'accent sur l'aspect incrémental du processus de classification automatique. Comme pour KBG, le résultat de Cobweb est une hiérarchie de classes. Mais cette hiérarchie est construite au fur et à mesure que les observations sont disponibles. À chaque instant, elle est disponible, et chaque nouvelle donnée y est incorporée à son tour. Le problème est donc ici de mettre a jour une hiérarchie de classes, de façon à prendre en compte une nouvelle observation. Cobweb le résoud en parcourant la hiérarchie du haut en bas, déterminant à chaque niveau la sous-classe la plus appropriée (cet algorithme est décrit plus complètement ci-dessous). Cet algorithme est essentiellement similaire à celui employé par l'algorithme UNIMEM.

Pour ce qui est de la forme des données traitées et des classes crées, Cobweb s'en tient, dans sa description originale, à un formalisme strictement propositionnel (dont les variables sont soit catégoriques, soit numériques). Différentes extensions ont été développées par la suite. Toutes ces extensions visent à permettre à Cobweb de manipuler des données structurées, à savoir des données décrites à différents niveaux de détail. Ce formalisme structuré, qui est intermédiaire entre un formalisme propositionnel et un formalisme relationnel, sera détaillé par la suite.

4.3 Cobweb et les données structurées

4.3.1 Principe de l'algorithme

La figure 4.6 schématise le fonctionnement de Cobweb. Une nouvelle observation est placée dans la hiérarchie à partir de la racine. À chaque niveau, la décision s'opère en considérant les partitions potentielles (générées par incorporation de l'objet dans chacune des sous-classes au niveau courant), qui sont évaluées. La meilleure des

partitions détermine vers quelle sous-classe immédiate l'objet sera dirigé. Suite à quoi, un appel récursif permet de poursuivre la progression de l'objet. Les cas dégénérés (une hiérarchie vide, ou restreinte à une seule classe) sont résolus trivialement.

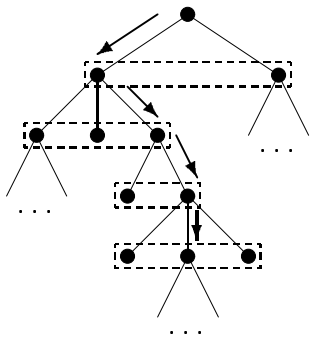


FIG. 4.6 – *Schéma de l'algorithme Cobweb. Chaque nouvelle observation est placée dans la hiérarchie existante, en partant de la racine. À chaque niveau, la partition courante est éventuellement réorganisée.*

Une telle stratégie, bien qu'efficace, ne permet cependant pas de construire des hiérarchies autres que binaires. Pour remédier à cet inconvénient, Cobweb teste un certain nombre d'alternatives à chaque niveau. En plus de chercher à placer l'objet dans les classes existantes, certaines modifications topologiques de la hiérarchie sont évaluées, et effectuées si elles conduisent à une meilleure partition. L'algorithme inclut trois opérateurs de restructuration, qui sont la création d'une nouvelle classe dans la partition courante, la fusion de deux classes, et le remplacement d'une classe par l'ensemble de ses sous-classes. Ces opérateurs permettent de déterminer automatiquement la structure de la hiérarchie, qui n'est donc pas toujours binaire.

La principale originalité de Cobweb, à ce stade, est de ne pas employer une mesure de similarité. Bien sûr, il remplace cette notion par une mesure de *densité* d'une classe. L'heuristique de classification que l'algorithme emploie ne considère que la partition générée : son but est de déterminer pour quelle partition le gain de densité entre une classe C et ses sous-classes $C_1, \ldots, C_K$ est maximal. Il suffit donc de générer les partitions potentielles, puis de les évaluer. Cette approche a l'avantage de permettre l'incorporation de classes déjà formées aussi bien que l'incorporation de nouvelles observations.

4.3.2 Structuration des données

La formation de concepts est un processus algorithmiquement efficace, qui résoud un problème crucial d'acquisition de connaissances. Puisque son fonctionnement est satisfaisant lorsque les données sont simples (c'est-à-dire se présentent sous la forme d'un vecteur de valeurs), il est intéressant d'étendre le même mécanisme d'apprentissage à des données complexes. Par « complexe», nous entendons des données dont la structure est hiérarchisée (à la manière des structures de données de l'algorithmique classique). Les niveaux de description intermédiaires (appelés dans la suite *niveaux d'abstraction*) permettent d'entrer progressivement dans le détail de la description de l'objet.

Dans ce résumé, nous ne décrirons qu'une modalité d'abstraction, définie par la notion d'*attribut multivalué*. Dans ce cas, l'un des attributs servant à décrire un objet a pour valeur un ensemble d'objets. Une telle description peut s'écrire :

type V = **tuple** (... a : **ensemble**(W), ...)

dans un hypothétique langage de description des données. Dans cet exemple, W est lui-même un type décrit par un tuple. Un type tel que V ne peut pas être mis à plat, c'est-à-dire être transformé en un type fondé uniquement sur le constructeur de tuples et les types primitifs.

4.3.3 Stratégie d'apprentissage

Lorsque les données sont structurées, les classes induites doivent dans une certaine mesure respecter cette structure. Imaginons ce que devrait être la description de l'attribut a d'une classe regroupant des éléments du type V ci-dessus. C'est pour cette raison qu'il est nécessaire de définir une *stratégie d'apprentissage*, qui prenne en compte les différents niveaux d'abstraction du domaine étudié. Une telle stratégie est fondée sur la remarque suivante : une classe d'objets abstraits (par exemple, des objets du type V) sera décrite par un certain nombre de références à des classes d'objets moins abstraits (par exemple, des classes regroupant des objets du type W).

Une telle organisation suppose que les classes soient générées par niveaux d'abstraction croissants. Dans l'exemple ci-dessus, les classes d'objets du type W doivent exister pour que les classes du type V puissent être définies. Intuitivement, il s'agit donc de « comprendre » le simple avant de comprendre l'abstrait. Le graphe d'abstraction (dont les sommets sont les types de données et les arcs les attributs) est donc parcouru en profondeur d'abord. L'algorithme d'apprentissage est alors constitué de

deux parcours imbriqués : le premier (dit *externe*) calqué sur la structure de l'espace d'abstraction, le second (ou *interne*) est un appel de Cobweb à chaque niveau d'abstraction. Incidemment, cette stratégie a l'inconvénient d'interdire les graphes d'abstraction cycliques : il est intéressant de faire le rapprochement entre cette limitation et une restriction de même nature, qui est communément admise dans les langages de représentation des connaissances.

4.3.4 Classes structurées

Nous avons décrit dans le paragraphe précédent de quelle façon l'apprentissage s'effectue dans un graphe d'abstraction. Il nous reste cependant à décrire de quelle façon Cobweb crée des hiérarchies de classes d'objets abstraits (par exemple, d'objets du type V). Nous avons vu que Cobweb génère un grand nombre de classes, qu'il décrit et évalue en termes de densité des valeurs couvertes. Dans cette section, nous allons décrire de quelle façon Cobweb décrit les classes qu'il génère, et comment cette description est évaluée.

Supposons une classe d'objets de type V (cf. exemple ci-dessus). Cobweb doit déterminer pour cette classe une decription intensionnelle (ou intentionnelle) de l'attribut multivalué a. Il dispose pour cela d'une hiérarchie de classes d'objets de type W, notée $\mathcal{H}_W$. Les classes de cette hiérarchie vont servir de « vocabulaire » à la description recherchée. Cette description, notée D, sera donc formée d'un ensemble de références à des classes de $\mathcal{H}_W$.

Cet ensemble de classes doit cependant respecter trois contraintes. Premièrement, tous les éléments (de type W) des valeurs de a (ces valeurs sont de type **ensemble**(W)) doivent être membre d'une (et d'une seule) classe référencée dans D. Deuxièmement, chaque classe de D doit avoir des membres dans chaque valeur de a. Enfin, les classes de D doivent être les plus spécifiques possibles. Si les classes référencées dans D respectent ces trois conditions, on peut dire que D forme un prototype des valeurs de a dans les objets couverts par la classe à caractériser. La figure 4.7 donne un exemple de description.

Nous ne détaillerons pas ici de quelle manière Cobweb évalue la densité d'une telle description : on peut cependant, intuitivement, comprendre que cette densité est, d'une certaine façon, liée à la profondeur (c'est-à-dire à la densité) des classes référencées dans la description. De plus, le calcul de la description d'un attribut multivalué peut se faire en un temps linéaire en la profondeur de la hiérarchie $\mathcal{H}_W$. Globalement, la complexité de l'algorithme d'apprentissage est de $O(N(\log N)^2)$ pour N objets de type V. Qui plus est, ce calcul est incrémental.

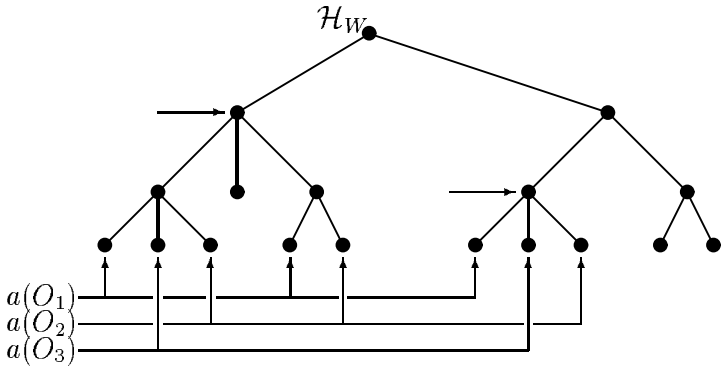


FIG. 4.7 – *Une caractérisation de a dans $\mathcal{H}_W$: sur cet exemple, les éléments des trois valeurs (notées $a(O_1)$, $a(O_2)$ et $a(O_3)$) sont localisés dans la hiérarchie. Les classes désignée par une flêche horizontale forment la description D de a dans la classe couvrant O_1, O_2 et O_3.*

4.3.5 Structure de la mémoire

La section précédente définit la représentation d'un attribut multivalué dans une classe regroupant des objets de type V. Le résultat de l'apprentissage n'est plus alors une simple hiérarchie de concepts, mais une structure complexe de hiérarchies enchevétrées. Il est intéressant alors de se poser la question de savoir quels liens entretiennent deux descriptions d'un attribut multivalué a dans des classes à deux niveaux différents. Notons C une classe de $\mathcal{H}_V$ (la hiérarchie des classes d'objets abstraits), et C' une sous-classe de C. Notons D et D' les descriptions de a dans C et C', respectivement.

Il est facile de voir alors que les classes référencées dans D' sont des sous-classes de celles référencées dans D : les hiérarchies $\mathcal{H}_V$ et $\mathcal{H}_W$ évoluent donc « parallèllement ». De plus, on peut montrer que toute classe référencée dans D est soit spécialisée (c'est-à-dire remplacée par une de ses sous-classes, directe ou non) dans D', soit différenciée (c'est-à-dire remplacée par plusieurs de ses sous-classes). L'algorithme détermine donc pour $\mathcal{H}_V$ une structure complexe, dont la construction n'est pas sans rappeler les hiérarchies de concepts formulables dans certains formalismes de représentation des connaissances.

4.4 Applications

4.4.1 Analyse d'images de télédétection par satellite

Notre domaine d'application privilégié est l'analyse automatique d'images de télédétection par satellite. Le but est ici de construire, à partir des données recueillies sur une image digitale, une base de connaissances dont les concepts représentent des constantes dans l'organisation spatiale de régions radiométriquement homogènes. Les données de la base sont donc des *paysages*, chaque paysage étant un morceau de l'image initiale, formé d'un ensemble de régions, chacune de ces régions étant décrite par un ensemble d'informations radiométriques.

On a donc le « schéma » suivant :

$$\begin{array}{lcl} \textbf{type } \textit{paysage} & = & \textbf{tuple } (\ a : \textbf{ensemble}(\textit{région})\ ) \\ \textbf{type } \textit{région} & = & \textbf{tuple } (\ \mathrm{xs}_1 : \textbf{num}, \mathrm{xs}_2 : \textbf{num}, \ldots\ ) \end{array}$$

(xs_1, xs_2, ... étant les valeurs – de type numérique – fournies par le satellite). Un jeu de données typique se présente de la façon suivante :

$$\begin{array}{lcl} \textbf{région } R_1 & = & (\mathrm{xs}_1 = 17, \mathrm{xs}_2 = 129, \ldots) \\ \ldots & & \\ \textbf{paysage } P_1 & = & (a = \{R_1, R_2, \ldots\}) \\ \textbf{paysage } P_2 & = & (a = \{R_7, R_2, R_4, \ldots\}) \\ \ldots & & \end{array}$$

On remarquera que le même élément peut apparaître dans plusieurs valeurs de a, ce qui signifie que les paysages se recouvrent partiellement.

Nos expérimentations ont porté sur un jeu d'essai de ce type, composé de plus de 46000 paysages, comprenant chacun entre 2 et 32 régions (avec une moyenne de 14,2 régions par paysage). Le résultat de l'apprentissage, obtenu après environ une heure de calculs, a permis de produire une carte thématique de la zone étudiée. Cette carte a été jugée très informative par l'expert. De plus, la description de chaque classe de paysage (composée de références à des classes radiométriques crées dans le même temps) a été reconnue parfaitement adaptée à la tâche.

4.4.2 Découverte de connaissances dans les bases de données

Plus généralement, les mécanismes de formation de concepts structurés s'appliquent dans un contexte de *découverte de connaissances dans les bases de données*,

et plus particulièrement pour la *fouille de données*, dont la problématique est d'extraire des connaissances non triviales de recueils de données. Les contraintes dans ce contexte sont de deux types.

Premièrement, les jeux de données sont massifs : on parle fréquemment de giga-, voire de téra-octets. Pour traiter raisonnablement de telles masses de données, il est nécessaire de disposer d'algorithmes efficaces. On considère en général qu'une complexité supérieure à $O(N^2)$ n'est plus tolérable. Les algorithmes de formation de concepts structurés répondent positivement à une telle contrainte. De plus, étant incrémentaux, ils ne sont pas sujets aux limitations imposées par la taille de la mémoire vive.

La seconde contrainte est qu'il s'agit de traiter des bases de données, lesquelles sont en général structurées selon un schéma. Ce schéma résulte d'une analyse préalable du domaine, et décrit la forme que prennent les données. Les formalismes de modélisation des données sont nombreux, mais une part importante d'entre eux intègrent, par exemple, la notion d'association multiple, qui est essentiellement équivalente à ce que nous avons appelé attribut multivalué. La formation de concepts structurés permet donc de tenir compte du schéma de la base de données, sans recourir ni à l'échantillonnage, ni à la reformulation, ce qui en fait un véritable outil de fouille des données.

4.5 Autres aspects de la formation de concepts structurés

Avant de terminer ce résumé, nous allons brièvement évoquer trois autres caractéristiques de la formation de concepts structurés. Deux de ces aspects concernent la modélisation des données à traiter. Tout d'abord, il est bien sûr possible de définir des *attributs simplement structurés*, c'est-à-dire des attributs dont la valeur est un (unique) sous-objet. On dispose donc d'une construction supplémentaire, illustrée par :

type V = **tuple** (... a : W, ...)
type W = **tuple** (...)

Ce cas se ramène au cas des attributs multivalués, l'attribut a ayant, dans ce cas, toujours un singleton pour valeur. Cependant, il est possible dans ce cas particulier de simplifier l'algorithme de calcul d'une description, et de réduire ainsi la complexité algorithmique de la classification.

De plus, et c'est là la deuxième caractéristique, les attributs simplement structurés offrent un moyen simple de *combiner des résultats de classification*. Supposons que plusieurs regroupements hiérarchiques aient été construits dans un domaine donné, à partir de données soit semblables soit distinctes dans chaque cas. Toute nouvelle donnée peut alors être considérée selon chacune de ces classifications. De fait, les nouvelles données peuvent être vues comme des abstractions de leurs différentes interprétations, et donc peuvent être reclassifiées comme objets structurés plutôt que selon leur description initiale.

Enfin, la dernière modalité de structuration des données concerne la notion d'ordre. La formation de concepts structurés s'appliquent à des données décrites selon le schéma suivant :

type V = **tuple** (... a : **liste**(W), ...)

On trouve ici une structure sensiblement similaire au cas des attributs multivalués, sinon que les valeurs de l'*attribut séquentiel* a sont ordonnées. Curieusement, cette contrainte additionnelle, qui semble mineure, complique considérablement le calcul d'une description pour un attribut séquentiel : une telle description est formée d'une liste de références à des classes de W, mais il doit exister une mise en correspondance surjective de chaque valeur vers la description, mise en correspondance qui doit respecter l'ordre des éléments des données. Un calcul exact est impossible, mais une solution heuristique peut être déterminée tout en conservant la complexité algorithmique du processus sous la barre de $O(N^2)$ et en maintenant l'aspect incrémental de l'algorithme.

4.6 Conclusion

Nous avons voulu, dans ce résumé donner un aperçu des possibilités offertes par la formation de concepts structurés. En plus d'offrir un support pour les modèles de données, cette technique permet la formation efficace de bases de connaissances fortement structurées. Elle forme donc un point de passage entre l'apprentissage et la représentation des connaissances complexes.

Bibliographie

[Bis 93] G. Bisson. *Induction de bases de connaissances en logique des prédicats.* Thèse de doctorat, Université de Paris-Sud, 1993. Cette thèse décrit un algorithme (appelé KBG) capable d'apprentissage non-supervisé lorsque les données et les concepts sont exprimés dans une variante de la logique du premier ordre. C'est à notre connaissance la tentative la plus avancée de combinaison des techniques de classification automatique (d'obédience statistique) et de représentation logique des connaissances.

[Fis 87] D. H. Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine learning*, 2:139–172, 1987. Cet article sert de fondement à tout notre travail. Fisher y décrit un algorithme permettant de construire une hiérarchie de concepts à partir d'une séquence d'observations. Cet algorithme (COBWEB) ne s'applique cependant qu'à des données décrites par des attributs catégoriques (nominaux).

[Fis 91] D. H. Fisher, M. J. Pazzani, et P. Langley, éditeurs. *Concept formation: knowledge and experience in unsupervised learning.* Morgan Kaufmann, 1991. Un ouvrage de synthèse sur la recherche en formation de concepts. Il vaut surtout par la variété des approches qui y sont présentées, ainsi que par (pour ce qui nous concerne) un chapitre décrivant une extension de COBWEB aux données structurées (reprenant le contenu de l'article suivant).

[Gen 89] J. H. Gennari, P. Langley, et D. H. Fisher. Models of incremental concept formation. *Artificial intelligence*, 40:11–61, 1989. Reprenant l'essentiel de l'article précédent, les auteurs fournissent cependant une intéressante perspective sur la recherche en apprentissage non supervisé. Une extension de COBWEB aux attributs numériques est décrite, et les prémisses d'un

algorithme de formation de concepts (c'est-à-dire de classification conceptuelle incrémentale et hiérarchique) à partir de données structurés y sont évoquées.

[Ket 95] A. Ketterlin. *Découverte de concepts structurés dans les bases de données.* Thèse de doctorat, Université Louis Pasteur, Strasbourg I, 1995. Cette thèse décrit dans le détail les mécanismes de formation de concepts structurés, selon trois modalités de structuration : les attributs simplement structurés, les attributs multi-valués, et les attributs séquentiels. On y trouve diverses applications de ces mécanismes, ainsi qu'un chapitre consacré à la découverte de connaissances dans les bases de données, qui traite du problème de la transformation d'un modèle conceptuel en un espace d'abstraction.

[Kod 95] Y. Kodratoff et M. Moulet. Découverte de connaissances dans les bases de données : présentation et état de l'art. *Actes des cinquièmes journées nationales PRC-GDR Intelligence Artificielle*, pages 283–296. Teknea Editions, 1995. Cet article est exactement conforme à ce que son titre suggère. Plus un programme qu'un rapport de recherche, il a le mérite de cerner la portée de la découverte de connaissances dans les bases de données, décrivant ainsi un certain nombre de directions d'investigation dans ce domaine.

[Leb 83] M. Lebowitz. Generalisation from natural language text. *Cognitive science*, 7:1–40, 1983. Un des premiers systèmes de classification conceptuelle (UNIMEM) est décrit dans cet article, qui est véritablement le premier exposé d'un système incrémental produisant une hiérarchie de concepts. Des travaux similaires ont été réalisés par Kolodner, et ont servi à fonder le raisonnement à partir de cas.

[Mic 83] R. S. Michalski et R. E. Stepp. Learning from observation: conceptual clustering. R. S. Michalski, J. G. Carbonell, et T. M. Mitchell, éditeurs, *Machine learning: an artificial intelligence approach (vol. I)*, chapitre 11, pages 331–363. Morgan Kauffman, 1983. Un des premiers articles sur l'apprentissage automatique non supervisé. Le système CLUSTER y est décrit. Les auteurs introduisent une terminologie (en particulier le terme de *classification conceptuelle*), et définissent une problématique générale permettant de distinguer les méthodes d'apprentissage non supervisé des méthodes de classification automatique.

[PS 91] G. Piatetsky-Shapiro et W. J. Frawley, éditeurs. *Knowledge discovery in databases*. AAAI/MIT Press, 1991. Un ouvrage regroupant une trentaine d'articles concernant la découverte de connaissances dans les bases de données et la fouille de données. Bien que datant un peu (uniquement en raison de l'affluence de publications sur ce sujet depuis lors), il reste un ouvrage de référence, puisqu'il définit cette nouvelle discipline. Signalons également la parution récente d'un nouvel ouvrage de synthèse qui fait suite à celui-ci.

[Tho 89] K. Thompson et P. Langley. Incremental concept formation with composite objects. *Machine learning: proceedings of the sixth international workshop*, pages 371–374. Morgan Kaufmann, 1989. Cet article décrit une première tentative d'étendre COBWEB aux objets et concepts structurés (ou *composites*). Utilisant un formalisme assez complexe et hétérogène (un objet est subdivisé en sous-objets, liés entre eux par des *relations*), l'algorithme comporte une phase de complexité exponentielle. C'est donc à la poursuite de ce travail que nous avons contribué.

Probabilités et réseaux de neurones

Jean-François Remm
CRIN-CNRS & INRIA-Lorraine
C.-él. : remm@loria.fr

5.1 Introduction aux probabilités

5.1.1 Événements et probabilités

Lors du lancer d'un dé, le résultat est une valeur parmi 1,2,3,4,5 ou 6. Mais on ne peut pas prévoir ce résultat plus précisément. On dit alors qu'on a effectué une *expérience aléatoire*. L'ensemble des résultats possibles, noté Ω (ici $\Omega = \{1, 2, 3, 4, 5, 6\}$) est appelé *univers*.
Chaque situation ou *événement aléatoire* (par exemple « obtenir un 2 », « obtenir un nombre impair ») qui est associée à cette expérience aléatoire peut s'exprimer à l'aide d'éléments de Ω (par exemple l'événement « multiple de 3 » est réalisé par les éventualités 3 et 6.). Tout événement aléatoire est donc assimilable à une partie de Ω.

Définition 1
Supposons que dans une expérience aléatoire, il y ait un nombre fini n d'éventualités, notées $e_1, e_2, \ldots e_n$.
On appelle *événement* toute partie de l'ensemble des éventualités $\Omega = \{e_1, e_2, \ldots e_n\}$ (appelé *univers*) .
Un événement est dit *élémentaire* s'il est constitué d'un seul élément.

Si l'on dispose d'un dé non pipé, le hasard intervient de manière uniforme en donnant autant de chances de se produire à chacun des résultats possibles. Intuitivement, l'événement « multiple de 3 » a 2 chances sur 6 de se produire et c'est par ce rapport 2/6 que l'on caractérise sa *probabilité* d'apparition. On définit une probabilité en associant à chaque événement aléatoire un nombre de l'intervalle $[0, 1]$

Définition 2
Si Ω est l'ensemble des éventualités associées à une expérience aléatoire, on appelle *probabilité* (ou *loi de probabilité*) sur Ω toute application P de $\mathcal{P}(\Omega)$ dans $[0, 1]$ telle

que $\forall (A, B) \in \mathcal{P}(\Omega) \times \mathcal{P}(\Omega)$:

$$\mathtt{P}(\Omega) = 1 \tag{5.2}$$

$$A \cap B = \emptyset \Rightarrow \mathtt{P}(A \cup B) = \mathtt{P}(A) + \mathtt{P}(B) \tag{5.3}$$

Le couple $(\Omega, \mathtt{P})$ est appelé *espace probabilisé*.

On note $\overline{A}$ l'événement contraire de l'événement A, c'est à dire son complémentaire dans Ω.
Deux événements aléatoires A et B disjoints ($A \cap B = \emptyset$) sont dits *incompatibles*.
Les ensembles Ω et $\emptyset$ sont respectivement appelés *événement certain* et *événement impossible*.

Propriété 1
Si $(A_1, A_2, \ldots, A_n$ incompatibles 2 à 2) alors $\mathtt{P}\left(\bigcup_{k=1}^{n} A_k\right) = \sum_{k=1}^{n} \mathtt{P}(A_k)$

Démonstration
on démontre par récurrence que la propriété (5.3) de la définition se généralise ainsi. □

Propriété 2
$\mathtt{P}(\overline{A}) = 1 - \mathtt{P}(A)$

Démonstration
en effet $\mathtt{P}(A) + \mathtt{P}(\overline{A}) = \mathtt{P}(A \cup \overline{A}) = \mathtt{P}(\Omega) = 1$. En particulier $\mathtt{P}(\emptyset) = 0$. □

Propriété 3
Si $B \subseteq A$ alors $\mathtt{P}(A - B) = \mathtt{P}(A) - \mathtt{P}(B)$

Démonstration
on a $\mathtt{P}(A - B) + \mathtt{P}(B) = \mathtt{P}((A - B) \cup B) = \mathtt{P}(A)$. □

Propriété 4
$\mathtt{P}(A \cup B) = \mathtt{P}(A) + \mathtt{P}(B) - \mathtt{P}(A \cap B)$

Démonstration
puisque $(A \cup B) - A = B - A = B - (A \cap B)$. Et donc en utilisant la propriété 3 on obtient que $\mathtt{P}(A \cup B) - \mathtt{P}(A) = \mathtt{P}(B) - \mathtt{P}(A \cap B)$ □

Propriété 5
Si $\{A_1, A_2, \ldots, A_n\}$ est un système complet (i.e. une partition de Ω) alors $\forall B \subset \Omega, \quad \mathtt{P}(B) = \sum_{k=1}^{n} \mathtt{P}(B \cap A_k)$

Démonstration
$\forall B \subset \Omega, \quad B = B \cap \Omega = B \cap \left(\bigcup_{k=1}^{n} A_k \right) = \bigcup_{k=1}^{n} (B \cap A_k)$. Puis il suffit d'utiliser la propriété 1. □

Remarque 3
Il est clair que tout événement aléatoire étant une réunion d'événements élémentaires (incompatibles par définition), le choix des $p_i = \mathtt{P}(\omega_i)$ pour tous les ω_i de Ω détermine entièrement l'application $\mathtt{P}$ puisque :

$$(\forall E \in \mathcal{P}(\Omega)) \quad \mathtt{P}(E) = \mathtt{P}\Big(\bigcup_{\omega_i \in E} \{\omega_i\} \Big) = \sum_{\omega_i \in E} \mathtt{P}(\{\omega_i\}) = \sum_{\omega_i \in E} p_i \tag{5.4}$$

5.1.2 Probabilité conditionnelle

L'idée de base permettant la compréhension de la notion de probabilité conditionnelle est la suivante. Si on dispose d'une information supplémentaire concernant l'expérience cela va changer la vraisemblance que l'on accorde *a priori* à l'événement étudié.

Si nous reprenons notre problème de jet de dé :

$$\begin{aligned} A = \text{« avoir un 2 »} = \{2\} \qquad & \mathtt{P}(A) = \frac{1}{6} \\ B = \text{« avoir un nombre pair »} = \{2,4,6\} \qquad & \mathtt{P}(B) = \frac{1}{2} \\ (A \text{ sachant } B) = & \\ \text{« avoir un 2 sachant qu'on a un nombre pair »} \qquad & \mathtt{P}(A \text{ sachant } B) = \frac{1}{3} \end{aligned}$$

Définition 3
Soit $(\Omega, \mathtt{P})$ un espace probabilisé et A un événement tel que $\mathtt{P}(A) > 0$. L'application $\mathtt{P}_A$ définie sur $\mathtt{P}(\Omega)$ par $\mathtt{P}_A(B) = \frac{\mathtt{P}(A \cap B)}{\mathtt{P}(A)}$ est une probabilité appelée *probabilité conditionnelle* (à la réalisation de A).
On note généralement $\mathtt{P}(B \mid A)$ au lieu de $\mathtt{P}_A(B)$ et on lit : probabilité de B sachant A.

La formule $\mathtt{P}(A \cap B) = \mathtt{P}(B \mid A)\mathtt{P}(A)$ est appelée *formule des probabilités composées.*
On vérifie que $\mathtt{P}(\bullet \mid A)$ est bien une probabilité au sens de la définition 2.

Remarque 4
On sait que si A est réalisé, B ne peut l'être que par les éléments ω de $A \cap B$; l'ensemble des résultats possibles n'est plus Ω mais A et les événements aléatoires sont maintenant les parties de A (notons que $\mathcal{P}(A) \subseteq \mathcal{P}(\Omega)$).

Propriété 6
$\mathtt{P}(B \mid A)\mathtt{P}(A) = \mathtt{P}(A \mid B)\mathtt{P}(B)$.

Démonstration
Conséquence immédiate de la symétrie de la formule des probabilités composées. □

Propriété 7 (Formule de Bayes)
Dans le cas où $\{A_1, A_2, \ldots, A_n\}$ est un système complet (i.e. une partition de Ω) :

$$\mathtt{P}(A_j \mid B) = \frac{\mathtt{P}(B \mid A_j)\mathtt{P}(A_j)}{\sum_{k=1}^{n} \mathtt{P}(B \mid A_k)\mathtt{P}(A_k)} \tag{5.5}$$

Démonstration
D'après la propriété 6 on a :

$$\mathtt{P}(A_j \mid B) = \frac{\mathtt{P}(B \mid A_j)\mathtt{P}(A_j)}{\mathtt{P}(B)}$$

donc en utilisant la propriété 5 on obtient bien le résultat. □

5.1.3 Variables aléatoires, espérance, variance

La notion de variable aléatoire formalise la notion de grandeur numérique dont la valeur dépend de l'issue d'une l'expérience aléatoire. Dans l'exemple du lancer de dés, si l'on considère une expérience aléatoire constituée d'une suite de lancers indépendants de deux dés, on peut s'intéresser à plusieurs variables aléatoires : $X_1(\omega) = \{$somme des chiffres obtenus au lancer des deux dés$\}$, $X_2(\omega) = \{$numéro du lancer correspondant à la première apparition d'un double six$\}$, ... Une variable aléatoire réelle est une valeur numérique réelle déterminée par le résultat d'une expérience aléatoire. C'est donc une application de Ω dans $\mathbb{R}$. On représente une variable aléatoire par une lettre majuscule telle X et sa réalisation x par une lettre minuscule.

Définition 4
On appelle *variable aléatoire (v.a.)* associée à un espace probabilisé $(\Omega, \mathtt{P})$ toute application définie sur Ω.

Définition 5
La *loi de probabilité* de X est la fonction qui à tout élément x fait correspondre la probabilité que x prenne cette valeur.

Définition 6
Soit X une variable aléatoire réelle définie sur $(\Omega, \mathtt{P})$ et dont la loi de probabilité est : $\{(x_i, p_i) \quad i \in \{1, \ldots, k\}\}$. On appelle *espérance mathématique* de X la valeur (si elle existe) :

$$E(X) = \sum_{i=1}^{k} x_i \, p_i = \sum_{i=1}^{k} x_i \, \mathtt{P}(X = x_i)$$

Cette formule correspond à une moyenne arithmétique des différentes valeurs de X pondérées par leurs probabilités.

Définition 7
On appelle *variance* de X, noté $var(X)$ ou σ^2 la quantité définie par :

$$\sigma^2 = E[(X - E(X))^2]$$

La variance est une mesure de la dispersion de X autour de $E(X)$.

5.2 Théorie Bayésienne de la décision

5.2.1 Introduction

Le but de la classification est de déterminer à partir d'une donnée (ou observation, ou exemple) X sa classe d'appartenance $C(X)$. Par exemple, les classes dans le cas d'une reconnaissance de caractères sont les différentes lettres de l'alphabet.

Cette attribution d'une classe à une observation se fait grâce à des fonctions dites discriminantes. La théorie bayésienne de la décision consiste à construire des fonctions discriminantes probabilistes.

On appelle probabilité *a posteriori* la probabilité conditionnelle $\mathtt{P}(C_i \mid x)$ de la classe C_i connaissant x. En d'autres termes, quand on a une entrée x, $\mathtt{P}(C_i \mid x)$ représente la probabilité que x appartienne à la classe C_i.
On appelle probabilité *intra-classes* (ou probabilité *a priori*) la probabilité conditionnelle $\mathtt{P}(x \mid C_i)$. Cela représente la probabilité d'avoir x comme entrée quand on sait que l'on est dans la classe C_i.

le classifieur assigne l'observation $X = x$ à la classe C_i (parmi M classes) si :

$$\mathtt{P}(C_i \mid x) = \max_{j=1\ldots M} \mathtt{P}(C_j \mid x)$$

En utilisant la formule de Bayes (voir propriété 7), le classifieur associe alors à x la classe C_i si :

$$\frac{\mathtt{P}(x \mid C_i)\mathtt{P}(C_i)}{\mathtt{P}(x)} = \max_{j=1\ldots M} \frac{\mathtt{P}(x \mid C_j)\mathtt{P}(C_j)}{\mathtt{P}(x)}$$

la règle de décision est alors :

$$\text{Classe(x)} = C_1 \text{ si } \mathtt{P}(X = x \mid C_1)\mathtt{P}(C_1) > \mathtt{P}(X = x \mid C_2)\mathtt{P}(C_2) \tag{5.6}$$

et on note :

$$\frac{\mathtt{P}(x \mid C_1)}{\mathtt{P}(x \mid C_2)} \underset{C_2}{\overset{C_1}{\gtrless}} \frac{\mathtt{P}(C_2)}{\mathtt{P}(C_1)} \tag{5.7}$$

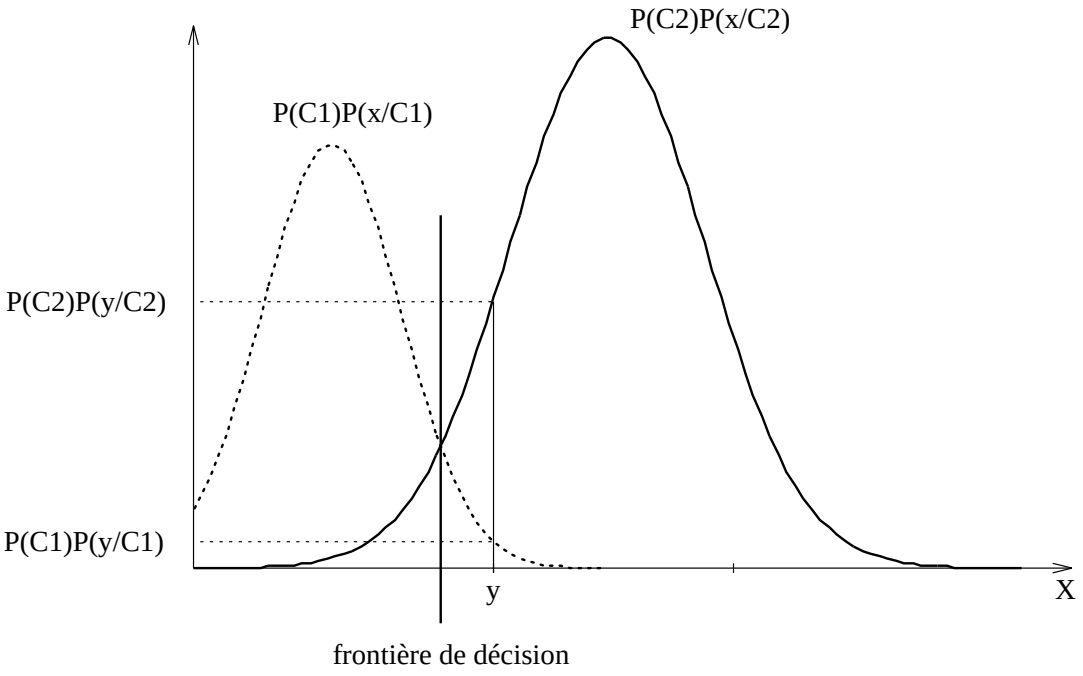


FIG. 5.8 – *Exemple à deux classes*

Le terme $\frac{\mathtt{P}(C_2)}{\mathtt{P}(C_1)}$ est appelé *rapport de vraisemblance* et la règle (5.7), *test du rapport de vraisemblance*. Le problème qui apparaît maintenant est l'estimation des probabilités intra-classes. Ces probabilités sont estimées indépendamment pour chaque classe.

5.2.2 Erreur de Bayes

Si on classe x dans C_1, la probabilité que x soit dans la bonne classe est donnée par le produit $\mathtt{P}(X = x \mid C_1)\mathtt{P}(C_1)$. La probabilité d'erreur est de même $\mathtt{P}(X = x \mid C_2)\mathtt{P}(C_2)$. Pour l'ensemble des observations, l'erreur totale correspond à la somme des deux aires ε_1 et ε_2 pour la règle de décision (5.6) (voir figure 5.2.1). Cette erreur se compose de deux termes, le premier dû aux éléments de L_1 attribués à la classe C_1, d'où l'erreur :

$$\varepsilon_1 = \mathtt{P}(C_2) \int_{L_1} \mathtt{P}(x \mid C_2) dx$$

le deuxième dû aux éléments de L_2 attribués à la classe C_2, d'où l'erreur :

$$\varepsilon_2 = \mathtt{P}(C_1) \int_{L_2} \mathtt{P}(x \mid C_1) dx$$

la probabilité d'erreur de classement $\varepsilon = \varepsilon_1 + \varepsilon_2$, est appelé *erreur de Bayes*. Elle est minimisée par la règle de décision précédente (voir équation (5.7)).

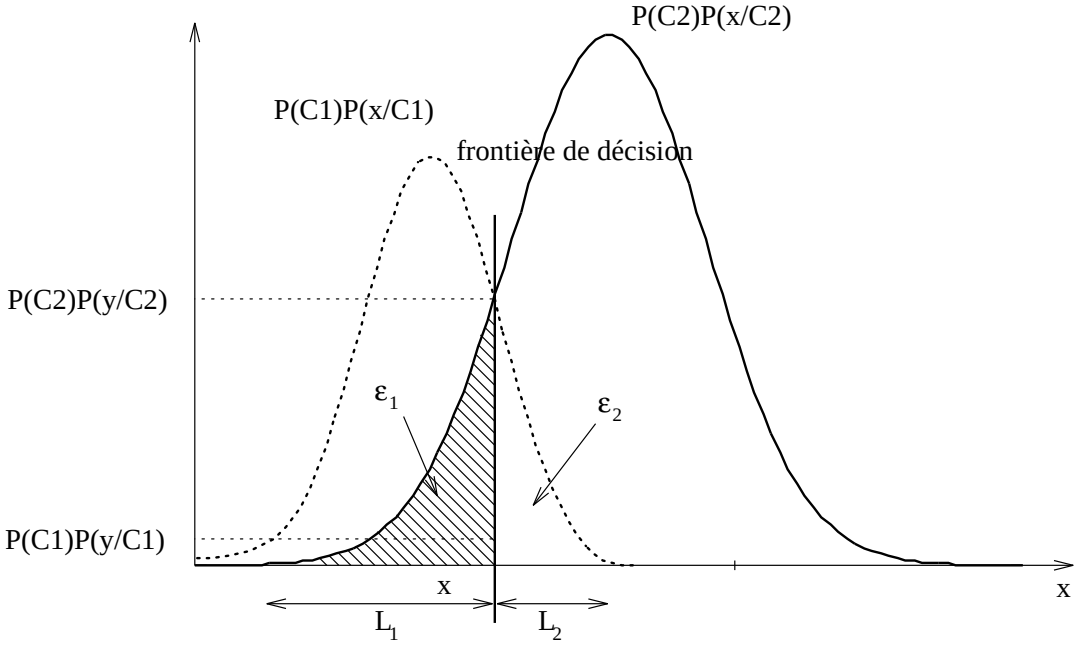


FIG. 5.9 – *Exemple à deux classes*

5.2.3 Risque bayésien

La minimisation de la probabilité d'erreur ε suppose que les pondérations des erreurs ε_1 et ε_2 sont semblables. En pratique, elle peuvent être différentes. Par exemple, il peut être plus risqué de considérer un malade comme bien portant que de classer une personne en bonne santé comme étant malade. On affecte donc à chaque situation un coût d'affectation C_{ij}.

$$C_{ij} = \text{coût de la décision } \{x \in C_i \text{ quand } x \in C_j\}$$

Le coût dû à l'affectation x dans la classe C_i est :

$$r_i(x) = \sum_{k=1}^{K} C_{ki} \mathtt{P}(x.C_k) \tag{5.8}$$

Comme notre but est de minimiser le coût $r_i(x)$, la règle de décision adoptée est la suivante :

$$r_i(x) \underset{C_j}{\overset{C_i}{\gtrless}} r_j(x) \tag{5.9}$$

Ce qui s'exprime sous la forme :

$$\frac{\mathtt{P}(x \mid C_i)}{\mathtt{P}(x \mid C_j)} \underset{C_j}{\overset{C_i}{\gtrless}} \frac{[C_{ji} - C_{jj}]\mathtt{P}(C_j)}{[C_{ij} - C_{ii}]\mathtt{P}(C_i)} \tag{5.10}$$

La règle (5.10) est appelée *test Bayésien du coût minimum*. Elle correspond à une règle du type (5.7) avec un seuil décalé.

5.3 Estimation de probabilités par réseaux de neurones

Des rapports étroits existent entre un classifieur bayésien et certains modèles connexionnistes. Nous allons montrer que pour un problème de classification à M classes, un perceptron multi-couches avec M neurones de sorties utilisant la rétro-propagation du gradient et ayant comme sorties désirées un neurone à 1 indiquant la classe correcte et les autres à 0, réalisera une estimation des probabilités *a posteriori*.

Plus formellement, pour attribuer un vecteur X à une classe parmi M, un classifieur bayésien estimera les probabilités *a posteriori* $\mathtt{P}(C_i \mid X)$ pour chaque classe $C_i,\ i = 1, \ldots, M$ et assignera au vecteur X la classe de probabilité maximale.
Contrairement à la méthode bayésienne qui estimera les probabilités a posteriori à partir des probabilités *a priori*, le calcul par un perceptron sera direct.

Démonstration
Pour une erreur des moindres carrés, le réseau minimisera :

$$\Delta = E\left\{\sum_{i=1}^{M}[y_i(X) - d_i]^2\right\}$$

ce qui peut s'écrire :

$$\Delta = \int\left\{\sum_{i=1}^{M}[y_i(X) - d_i]^2\right\}\mathtt{P}(X)\,dX$$

On a $\mathtt{P}(X) = \sum_{i=1}^{M}\mathtt{P}(X.C_i)$, d'où :

$$\Delta = \int\sum_{j=1}^{M}\left\{\sum_{i=1}^{M}[y_i(X) - d_i]^2\right\}\mathtt{P}(X.C_j)\,dX$$

Étant donné que $\mathtt{P}(X.C_j) = \mathtt{P}(C_j \mid X)\mathtt{P}(X)$, nous obtenons :

$$\Delta = \int\left[\sum_{j=1}^{M}\sum_{i=1}^{M}[y_i(X) - d_i]^2\,\mathtt{P}(C_j \mid X)\right]\mathtt{P}(X)\,dX$$
$$\Delta = \int\sum_{k=1}^{M}\left[\sum_{j=1}^{M}\sum_{i=1}^{M}[y_i(X) - d_i]^2\,\mathtt{P}(C_j \mid X)\right]\mathtt{P}(X.C_k)\,dX$$

en utilisant le changement qui a permis de passer de la ligne 1 à la ligne 3 dans l'autre sens, nous obtenons :

$$\Delta = E\left\{\sum_{j=1}^{M}\sum_{i=1}^{M}[y_i(X) - d_i]^2\,\mathtt{P}(C_j \mid X)\right\}$$

puis en développant l'expression à l'intérieur des crochets :

$$\Delta = E\left\{\sum_{j=1}^{M}\sum_{i=1}^{M}\left[{y_i}^2(X)\mathtt{P}(C_j \mid X) - 2y_i(X)d_i\mathtt{P}(C_j \mid X) + {d_i}^2\mathtt{P}(C_j \mid X)\right]\right\}$$

en exploitant le fait que ${y_i}^2(X)$ n'est fonction que de X et que $\sum_{j=1}^{M} \mathtt{P}(C_j \mid X) = 1$ nous obtenons :

$$\begin{aligned}\Delta &= E\left\{\sum_{i=1}^{M}\left[{y_i}^2(X) - 2y_i(X)\sum_{j=1}^{M} d_i \mathtt{P}(C_j \mid X) + \sum_{j=1}^{M} {d_i}^2 \mathtt{P}(C_j \mid X)\right]\right\} \\ &= E\left\{\sum_{i=1}^{M}\left[{y_i}^2(X) - 2y_i(X)E\{d_i \mid X\} + E\{{d_i}^2 \mid X\}\right]\right\}\end{aligned}$$

où $E\{d_i \mid X\}$ et $E\{{d_i}^2 \mid X\}$ sont les probabilités conditionnelles respectivement de d_i et de ${d_i}^2$. En additionnant et soustrayant $\sum_{i=1}^{M} E^2\{d_i \mid X\}$ nous obtenons une expression bien plus significative.

$$\begin{aligned}\Delta &= E\left\{\sum_{i=1}^{M}\left[{y_i}^2(X) - 2y_i(X)E\{d_i \mid X\} + E^2\{d_i \mid X\}\right.\right. \\ &\qquad \left.\left. + E\{{d_i}^2 \mid X\} - E^2\{d_i \mid X\}\right]\right\} \\ &= E\left\{\sum_{i=1}^{M}[y_i(X) - E\{d_i \mid X\}]^2\right\} + E\left\{\sum_{i=1}^{M} var\{d_i \mid X\}\right\}\end{aligned}$$

Le second terme de l'expression est indépendant des sorties y_i du réseau. La minimisation de Δ est donc équivalente à minimiser le premier terme. Et pour un problème de classification 1 parmi M, d_i vaut 1 si l'entrée X appartient à la bonne classe, 0 sinon. Donc :

$$\begin{aligned}E\{d_i \mid X\} &= \sum_{j=1}^{M} d_j \mathtt{P}(C_j \mid X) \\ &= \mathtt{P}(C_i \mid X)\end{aligned}$$

ainsi, l'erreur quadratique est minimale pour un réseau estimant les probabilités *a posteriori*. □

Pour chaque type d'erreur : coût du maximum de vraisemblance, entropie relative, distance de Kullback-Leibler, ... usuellement utilisé, des démonstrations similaires existent [Mon 94, Ric 91, Ruc 90].

Bibliographie

[Bre 88] P. Bremaud. *Modélisation des phénomènes aléatoires.* Springer-Verlag, 1988. Exposé introductif simple, avec des exemples d'applications en théorie de l'information.

[Che 94] B. Cheng et D.M. Titterington. Neural networks: A review from a statistical perspective. *Statistical Science*, 9(1):2–54, 1994.

[Mon 94] Christophe Monrocq. *Approche probabiliste pour l'élaboration et la validation de système de décision; Application aux réseaux de neurones.* Thèse de Doctorat, Université Paris-Dauphine, 1994.

[Ric 91] Michael D. Richard et Richard P. Lippmann. Neural network classifiers estimate bayesian a posteriori probabilities. *Neural Computation*, 3:461–483, 1991.

[Rie] V. Ries. *Cours de probabilité.* cours de Deug.

[Ruc 90] D.W. Ruck et S.K. Rogers. The multilayer perceptron as an approximation to bayes optimal discriminant function. *IEEE Trans. Neural Networks*, 1:296–298, 1990.

[Sap 90] G. Saporta. *Probabilités, Analyse des données et Statistiques.* Éditions Technip, 1990.

Introduction au problème de l'interprétation des données en classification

Jean-Daniel Kant
CRIN-CNRS & INRIA-Lorraine
C.-él. : kant@loria.fr

6.1 Introduction

La *catégorisation* joue un rôle fondamental dans la représentation et l'utilisation des connaissances, que ce soit par un sujet humain, animal ou artificiel. Dans la langue française, catégorisation signifie le « regroupement d'objets de même nature dans une classe ». Elle intervient aussi bien chez l'homme pour faciliter un diagnostic, un raisonnement ou une décision; chez l'animal afin de reconnaître un signal; en reconnaissance des formes sur une machine pour classer et reconnaître des informations; on parle alors de classification automatique.

Dans cet exposé nous nous intéressons plus particulièrement aux systèmes de classification automatique (SCA). Dans un premier temps, nous tenterons de clarifier la distinction entre *catégorisation* et *classification*. Cela nous conduira ensuite à poser le problème du rôle des SCA à travers celui de l'interprétation des classifications, en particulier lorsqu'elles portent sur des données qualitatives et donc à fort contenu sémantique. Ce problème ayant été posé, nous l'étudierons pour 3 SCA : un réseau connexionniste, le perceptron multi-couches; un arbre de décision (C4.5); et enfin un réseau connexionniste d'inspiration psychologique, le réseau Categ_ART.

6.2 Interprétation et analyse des données en classification

6.2.1 Catégorisation et classification

L'activité de catégorisation fait appel à deux activités distinctes mais complémentaires :

- la *catégorisation* proprement dite telle que nous venons de la définir comme un *processus de formation de catégories*;

- la *classification* qui consiste, pour un objet donné, à déterminer la classe d'appartenance de cet objet et donc la catégorie qui le représente[2]. La classification fait donc intervenir un *processus de décision* d'appartenance.

Il est clair que ces deux activités sont liées et complémentaires. En phase de catégorisa-tion, il est souvent nécessaire de construire des fonctions de décision d'appartenance pour grouper un nouvel objet avec les objets d'une catégorie en cours de formation. De même la fonction ou règle de décision utilisée en classification dépend fortement des structures des catégories formées par la catégorisation.

Cependant, une différence majeure entre catégorisation et classification se situe dans la dynamique de ces activités : la catégorisation est un processus dynamique de construction de représentations et de structures catégorielles, cette activité allant souvent de pair avec un *processus d'apprentissage* des relations entre les objets à catégoriser. En revanche, la classification opère sur des structures *a priori* stabilisées (les catégories) et s'intéresse uniquement au problème d'appartenance d'un objet à une catégorie. Elle peut en particulier permettre une *généralisation des classements* effectués pendant la catégorisation, en classant un objet nouveau.

6.2.2 Interprétation des classifications

L'interprétation joue souvent un rôle fondamental en classification automatique. Comme le souligne Diday *et al.* [Did 82, p. 73], «l'information apportée par une classification se situe au niveau sémantique». Une «bonne» classification n'est pas une classification donnant un résultat vrai ou faux, probable ou improbable mais exploitable ou non exploitable, dans le but d'expliciter des relations nouvelles entre les données. L'interprétation peut intervenir à deux niveaux :

- au niveau de la catégorisation, en explicitant par exemple des structures catégorielles représentatives d'une catégorie (e.g. prototypes; Cf. section 6.3.3).

- au niveau de la classification (cas le plus fréquent), en explicitant les critères qui permettent de décider de l'appartenance d'un objet à une catégorie. On parle alors de *règles de décision*, qui permettent d'*expliquer* les catégorisations effectuées.

2. Dans notre vocabulaire, une *classe* est une sous-catégorie, au sens d'un sous-ensemble d'un ensemble. Une catégorie est donc constituée de l'union, au sens de l'opération d'union en théorie mathématique des ensembles, des classes qui sont ses sous-catégories.

Dans les SCA, l'interprétation peut s'effectuer de façon *corrélée* ou *décorrélée*. Dans le cas d'une interprétation corrélée, le SCA fournit explicitement des structures catégorielles et/ou des règles de décision interprétables par un utilisateur. Cela implique que le processus de catégorisation et/ou de classification soit fortement lié au processus d'interprétation : les algorithmes doivent satisfaire la contrainte : «des classifications sont interprétables». Dans le cas d'une interprétation décorrélée, le processus d'interprétation est indépendant du processus de catégorisation ou de classification. Dans certains, il peut être difficile à mettre en œuvre (e.g. perceptrons; Cf. section 6.3.1).

Il semble donc préférable d'utiliser des interprétations corrélées, de manière à s'assurer que l'interprétation est possible. Cela n'est cependant pas facile à réaliser, car les algorithmes utilisés par les SCA font généralement appel à des mesures de *similarité* entre les objets, souvent numériques. Le problème est donc de trouver la bonne mesure de similarité qui permette de rendre compte des phénomènes que l'on souhaite observer pendant l'interprétation. Dans le cas de données qualitatives, il faut ajouter la contrainte de respecter le caractères symbolique des données tout en utilisant des mesures numériques de similarité.

De façon générale, le problème de l'interprétation provient du fait que l'on cherche en général à expliquer des catégorisations à l'aide de règles de classification, alors que ces processus ne sont pas de même nature. On peut en effet souligner une autre distinction :

- la catégorisation consiste à *mettre ensemble des objets qui sont équivalents par rapport à un but ou un concept donné*. Il y a donc construction d'une relation d'équivalence (au sens non nécessairement mathématique de ce mot) ;
- la classification consiste plutôt à *mettre ensemble des objets sur la base de proximités définies à partir des données*. Il y a donc construction d'une mesure numérique de similarité, qui passe souvent par un recodage des données.

Le problème est donc de savoir dans quelle mesure le recodage et la similarité peuvent rendre compte de manière suffisamment explicite et interprétable des relations d'équivalence entre les objets.

6.2.3 Cas des méthodes « classiques » en analyse des données

Avant d'examiner plus en détails les 3 SCA choisis, nous pouvons examiner le problème de l'interprétation dans le cas de quelques méthodes utilisées classiquement

en analyse des données. De façon générale, on peut faire appel aux SCA pour réaliser deux types d'objectifs :

- *construction de classes* qui représentent le plus fidèlement possible des objets décrits par un tableau de données (typiquement matrice objets / propriétés (Cf. chapitre 1) ;
- *explication* de catégorisations effectuées par ailleurs.

Suivant les systèmes, seul l'un des deux objectifs est réalisé; parfois les deux.

Dans le cas de la *classification ascendante* et la *classification descendante* (nuées dynamiques) [Did 82, ch. 2], seule la construction des catégories est réalisée. Cependant, la classification ascendante offre l'avantage de fournir une distance entre les classes (indices de la hiérarchie).

L'*analyse discriminante* a pour but de réaliser l'explication. Chaque objet est décrit par n variables quantitatives ou qualitatives, et par une variable qualitative qui désigne la classe de l'objet. On cherche alors à construire uen fonction de décision qui effectue les classements avec un taux d'erreur le plus faible possible. On peut aussi chercher une bonne tolérance au bruit, de bonnes performances en généralisation etc. La classification bayésienne peut être utilisée en analyse discriminante. Cependant, toutes ces méthodes utilisent en général des fonctions de décision numériques et qui, si elles peuvent fournir de bons résultats en terme de performance (résistance au bruit, généralisation), ne sont donc pas facilement interprétables, notamment sous forme symbolique quand les données sont qualitatives. Notons que de façon générale, la plupart des méthodes d'analyse de données travaillent dans des espaces de représentations continus ($\mathbb{R}^n$) et sont donc plutôt adaptées à des données quantitatives.

6.3 Présentation de trois systèmes de classification automatique

Dans cette section, nous étudions trois types de SCA qui ont tous en commun de réaliser l'apprentissage de classifications déjà effectuées, contrairement aux algorithmes de classification ascendante et descendante qui forment des classes à partir de données. Ces trois SCA mettent en œuvre des algorithmes d'apprentissage supervisé afin de construire des fonctions de décision qui permettent de reproduire les classements appris et de les généraliser à des objets nouveaux. Nous étudierons

plus particulièrement les performances en terme de *description* et d'*explication*. Le pouvoir descriptif caractérise ici la capacité du SCA a construire des classifications fiables et d'obtenir de bons résultats en généralisation. Le pouvoir explicatif est la capacité de fournir des éléments facilement interprétables qui permettent d'expliquer les classifications effectuées (e.g. règles de décision).

6.3.1 Perceptron Multi-Couches

Le perceptron de Rosenblatt [Ros 58] est un des premiers modèles connexionnistes pour la classification (Cf. exposé de Frédéric Alexandre). A l'origine, il se compose de deux couches de neurones une d'entrée, représentée par le vecteur $\mathbf{x}$, qui code en quelque sorte la perception du réseau, et une de sortie, que nous représentons par le vecteur $\mathbf{y}$, et qui code la décision du réseau.

On a :

$$\begin{aligned} \forall j \in \{1, 2, ..., n\} \quad a_j &= \sum_{i=1}^{n} W_{ij} x_i \\ y_j &= \begin{cases} 1 & \text{si } a_j > \theta_j \\ 0 & \text{sinon} \end{cases} \end{aligned} \tag{6.11}$$

Si la sortie y_j est la fonction caractéristique de l'ensemble des éléments de la catégorie C_j (fonction d'appartenance), alors le perceptron peut servir à classer les objets dans les C_j, si celles-ci sont *linéairement séparables*. On utilise en général la règle d'apprentissage de Widrow-Hoff [Wid 60] en modifiant les poids des connexions de la façon suivante :

$$W_{ij}^{(t+1)} = W_{ij}^{(t)} + \eta(d_j - y_j)x_i \tag{6.12}$$

où $(d_j)_{i \in \{1,2,...,n\}}$ représente le vecteur de réponse désirée pour une entrée $\mathbf{x}$ donnée, c'est à dire la catégorie de $\mathbf{x}$, et η une constante entre 0 et 1 (voir par exemple [Abd 94, chap. II] pour une présentation plus détaillée). Dans le cas de classes non linéairement séparables, on ajoute des couches intermédiaires (couches cachées) entre l'entrée et la sortie. Les connexions sont calculées de façon itérative en appliquant *l'algorithme de rétropropagation du gradient* de l'erreur entre la sortie désirée et la sortie du réseau [LC 86], [Rum 86].

Concernant les capacités descriptives des perceptrons multi-couches (PMC), elles sont en théorie très bonnes, puisqu'il a été prouvé par Cybenko qu'un PMC pouvait approximer n'importe quelle fonction réelle continue avec un taux d'erreur très

faible [Cyb 89]. Un PMC est donc en théorie un « approximateur universel ». Le problème est qu'en pratique, les paramètres du PMC (constantes η, nombre de couches cachées) sont souvent difficiles à déterminer et le temps d'apprentissage est souvent très long en raison de la méthode itérative de gradient employée. Notons que le PMC ne fonctionne qu'en mode *supervisé*, c'est à dire qu'il nécessite de connaître à l'avance les catégories des objets à apprendre. Il n'en reste pas moins que les PMC sont à l'heure actuelle les modèles connexionnistes les plus employés, en reconnaissance de forme, en traitement d'image et de la parole notamment, offrant souvent de meilleures performances en terme d'apprentissage et de généralisation que les méthodes statistiques classiques. La performance est donc satisfaite (elle est même idéale en théorie, en vertu du théorème de Cybenko), en dépit d'une difficulté certaine de mise en œuvre.

En terme de traitement de l'information, un PMC est équivalent à un filtre non linéaire, dont la fonction de transfert est donnée par le produit des matrices des connexions inter-couches. L'algorithme de rétropropagation utilise une méthode d'optimisation (descente de gradient) pour calculer les paramètres de ce filtre (i.e. les poids des connexions) de manière à s'ajuster le mieux possible à la sortie désirée. De fait, le PMC peut apparaître comme une « boîte noire », qui calcule des corrélations non linéaires entre ses entrées et ses sorties.

En conséquence, l'inconvénient majeur du PMC est qu'il ne peut fournir directement des règles explicites de son fonctionnement, c'est à dire réaliser l'explication des catégorisations. Il a en effet été conçu pour classer des objets et non pas pour donner des règles de classification. Si on regarde la fonction de décision d'un perception simple (équation 6.11), on voit que l'appartenance à la catégorie C_j est déterminée en fonction de la position de l'entrée x par rapport à l'hyperplan d'équation :

$$\sum_{i=1}^{n} W_{ij} x_i - \theta_j = 0$$

Hormis cette formulation mathématique, il n'y a aucun moyen d'exprimer clairement une règle d'appartenance à la catégorie. Certains travaux récents tentent de remédier à ce problème en essayant d'extraire des règles logiques à partir des valeurs des connexions et des activités des couches du PMC. Parmi ces travaux, on peut citer l'exemple de l'extraction de traits sémantiques pour modéliser la première annonce de bridge, réalisée par Bochereau et Bourgine [Boc 89]. Comme le souligne ces auteurs, le processus d'extraction sémantique des règles n'est pas réalisé par le réseau, mais doit être effectué par des processus d'autres natures, souvent complexes. On doit alors, comme le remarquent Bochereau, Bourgine et Deffuant, « comparer le temps

de calcul global (construction du réseau + extraction) et les performances avec celui des méthodes qui produisent directement des règles. C'est le cas de l'apprentissage symbolique» ([Boc 91, p.155]).

En résumé, on voit donc que le PMC représente un modèle qui privilégie largement le pouvoir descriptif au détriment du pouvoir explicatif. La difficulté à interpréter un PMC en termes de fonctionnement symbolique peut s'avérer gênante dans le cas où les données sont de nature symbolique. Ceci provient du fait que *l'algorithme d'apprentissage et de fonctionnement* de ce réseau est de l'ordre de l'optimisation statistique et est donc totalement indépendant *a priori* d'un processus symbolique de classification.

6.3.2 Les arbres de décision

Les arbres de décision ont été introduits dans les années 80, notamment grâce aux travaux de Breiman et ses collègues, donnant le modèle CART [Bre 84] et ceux de Quinlan, donnant le modèle ID3 puis C4.5 [Qui 93]. Les arbres de décision sont des algorithmes d'apprentissage symbolique qui utilisent des mesures probabilistes pour apprendre des classifications en construisant des partitions arborescentes qui peuvent s'interpréter comme des arbres de décision. Ils offrent donc l'avantage d'allier un pouvoir descriptif à un bon pouvoir explicatif. Nous présenterons ici uniquement le modèle C4.5 de Quinlan, présenté en détail dans [Qui 93]. Il offre l'avantage d'être assez simple et générique, dans la mesure où la plupart des arbres de décision actuels lui sont proches.

Architecture d'un arbre de décision

Un arbre de décision a pour but d'expliciter une catégorisation dans le cas où les objets classés sont décrits par un ensemble de valeurs d'attributs qui représentent les propriétés caractéristiques des objets. Un arbre de décision est un arbre dont les noeuds sont :

- soit des *feuilles* contenant des objets appartenant tous à une même catégorie. Les feuilles représentent donc les classes d'une même catégorie C (sous-ensembles de C).

- soit des *noeuds de décision* qui partitionnent les données suivant plusieurs sous-ensembles, chaque sous-ensemble correspondant à un résultat d'une fonction de test, cette fonction caractérisant le noeud de décision.

Illustrons ceci sur un exemple en considérant le tableau 1 de données ci-dessous, qui indique si on peut jouer au golf compte tenu des conditions climatiques représentées par 2 variables qualitatives (aspect du ciel et vent) et 2 variables quantitatives (température et humidité).

outlook	temperature($^\circ$ F)	humidity (%)	Windy?	class
sunny	85	85	false	Don't Play
sunny	80	90	true	Don't Play
overcast	83	78	false	Play
rain	70	96	false	Play
rain	68	80	false	Play
rain	65	70	true	Don't Play
overcast	64	65	true	Play
sunny	72	95	false	Don't Play
sunny	69	70	false	Play
rain	75	80	false	Play
sunny	75	70	true	Play
overcast	72	90	true	Play
overcast	81	75	false	Play
rain	71	80	true	Don't Play

TAB. 6.1 – *Exemple de données (d'après [Qui 93]).*

Un moyen de représenter les informations du tableau 6.1 est l'arbre de décision représenté sur la figure 6.10.

Algorithme de construction

L'algorithme de construction d'un arbre de décision est du type « diviser et conquérir » (divide and conquer). Il s'agit de partitionner un ensemble E d'apprentissage en une partition $E_1, E_2, .., E_n$, où chaque E_i correspond à un résultat du test employé. La procédure est répétée récursivement sur toutes les sous-partitions jusqu'à ce que tous les nœuds terminaux ne contiennent que des objets d'une même classe. Dans les types d'arbre de décision que nous considérons, les tests portent uniquement sur un attribut à la fois : ils sont donc de la forme $X = a$ (pour des données

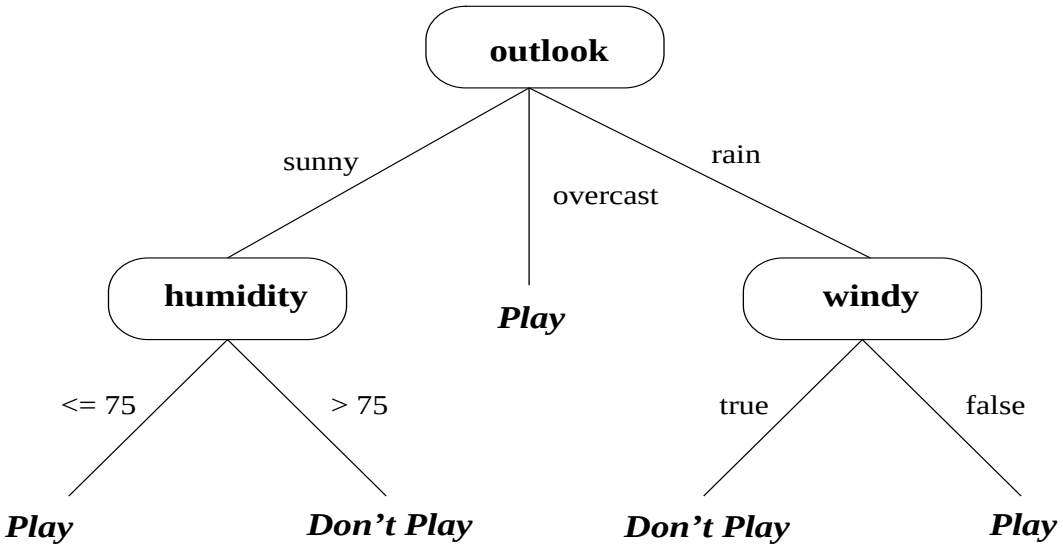


FIG. 6.10 – *Arbre de décision correspondant au tableau 6.1*

qualitatives) ou $X <= a$ pour des données quantitatives, X désignant un attribut et a une valeur.

Choix des tests

La construction de l'arbre se ramène donc à déterminer les tests à appliquer (quel attribut?, quelle valeur de comparaison?) et l'ordre dans lequel on effectue les tests (d'abord sur l'attribut «overlook» puis sur «humidity» ou l'inverse?), de manière à trouver l'arbre le plus petit possible. Pour ce faire, on ne peut qu'employer des heuristiques car le problème de déterminer l'arbre de décision le plus petit à partir d'un tableaux de données est NP-complet. La plupart des méthodes construisent des fonctions d'évaluation et choisissent le test courant qui donne la meilleure valeur.

Les algorithmes proposés par Quinlan (ID3, C4.5) utilisent des fonctions d'*entropie* fondées sur la théorie de l'information. Notons $F(C_j, S)$ le nombre d'éléments d'un ensemble S qui sont dans la classe C_j. La probabilité qu'un élément de S tiré au hasard soit dans C_j vaut $F(C_j, S)/|S|$, où $|S|$ est le cardinal de S. La quantité d'information d'une partition de S en k classes $C_1, C_2, .., C_k$ (entropie de S) vaut

donc :

$$info(S) = -\sum_{j=1}^{k} \frac{F(C_j, S)}{|S|} \times log_2(\frac{F(C_j, S)}{|S|}) \text{ bits.} \tag{6.13}$$

Si on effectue le même calcul après avoir appliqué un test X qui partitionne l'ensemble E d'apprentissage en une partition $E_1, E_2, .., E_n$, la quantité d'information de ce partionnement s'écrit :

$$info_X(E) = \sum_{i=1}^{n} \frac{|E_i|}{|E|} \times info(E_i) \tag{6.14}$$

L'algorithme ID3 utilise donc comme mesure du test X la fonction de *gain* définie par :

$$gain(X) = info(E) - info_X(E) \tag{6.15}$$

qui mesure la quantité d'information qu'apporte le partionnement du test X. Le critère utilisé par ID3 est celui de choisir à chaque étape le test donnant le gain maximum parmi tous les tests possibles (*gain criterion*). Cependant, le gain défini par les équations 6.13, 6.14 et 6.15 a l'inconvénient de favoriser les tests sur les attributs ayant un grand nombre de valeurs possibles. Pour corriger ce biais, C4.5 utilise une version normalisée du gain, le *gain ratio* :

$$gain\_ratio(X) = gain(X)/split\_info(X) \tag{6.16}$$

où la fonction de normalisation $split\_info(X)$ vaut

$$split\_info(X) = \sum_{i=1}^{n} \frac{|E_i|}{|E|} \times log_2(\frac{|E_i|}{|E|}) \tag{6.17}$$

Formes possibles de tests

Dans le cas d'attributs à *valeurs discrètes*, les tests seront de la forme $X_k = a$, où a est une valeur possible pour l'attribut X_k. Les attributs et leurs valeurs étant en nombre fini, l'ensemble de tous les tests possibles est fini et notre algorithme précédent fondé sur le gain peut fonctionner. A chaque étape, on crée autant de sous-branches du nœud courant de l'arbre qu'il y a de valeur possible pour l'attribut X_k sélectionné pour le test.

Dans le cas d'attributs à *valeurs continues*, les tests seront de la forme $X_k \leq \sigma$, ou $X_k > \sigma$ où σ est une valeur réelle seuil. Le nombre de valeurs seuils étant infini *a priori*, l'ensemble de tous les tests possibles n'est plus fini et notre algorithme précédent fondé sur le gain ne peut plus fonctionner tel quel. La solution consiste à déterminer un intervalle I_k des valeurs possibles pour l'attribut X_k, puis à discrétiser cet intervalle en m points $v_1, v_2, .., v_m$. On choisit alors des seuils σ_l tels que:

$$\sigma_l = \frac{v_l + v_{l+1}}{2}$$

Évaluation des arbres de décision

Les arbres de décisions sont des SCA attractifs car ils offrent l'avantage d'offrir à la fois un bon pouvoir explicatif (bonnes capacités de performances en classification et en généralisation, souvent du même niveau des meilleurs algorithmes de classification) et un bon pouvoir descriptif, dans la mesure où on peut interpréter l'arbre comme un ensemble de règles de décision. Il suffit pour cela de partir du sommet de l'arbre et de former tous les chemins possibles pour obtenir une feuille d'une classe. Ainsi, dans l'exemple du jeu de golf de la figure 6.10, on a comme règles :

classe = «Play» ssi (outlook = sunny et humidity $\leq$ 75) ou (outlook = overcast) ou (outlook = rain et windy = false)

classe = «Don't Play» ssi (outlook = sunny et humidity $>$ 75) ou (outlook = rain et windy = true)

On obtient donc des règles logiques, sous forme normale disjonctive (SI .. ALORS .. SINON...).

Cependant, les arbres de décision ont comme principal inconvénient d'être souvent grands, c'est à dire de posséder beaucoup de feuilles. Ceci est dû entre autres au fait que l'on crée un nœud de décision pour chaque résultat possible d'un test. Lorsque l'arbre est important, il devient illisible et l'on perd ainsi en pouvoir explicatif. Il faut alors recourir à des méthodes automatiques d'extraction de règles à partir de l'arbre. On trouvera une présentation de quelques méthodes d'extraction dans [Qui 93, ch. 5]. De plus, pour réduire la taille de l'arbre et améliorer les performances en généralisation, des méthodes d'élagage sont souvent appliquées (Cf. [Qui 93, ch. 4]).

6.3.3 Approche psychomimétique: Categ_ART

Pour terminer cet exposé, nous présentons un réseau connexionniste d'inspiration psychologique qui effectue l'apprentissage automatique de classifications et fournit directement un ensemble de règles logiques qui expliquent ces classifications. Ce réseau, Categ_ART, est présenté en détail dans [Kan 96]. Nous en donnons ici les grands principes.

Le réseau Categ_ART est un réseau connexionniste psychomimétique, ce qui signifie qu'il met en œuvre de façon effective un modèle cognitif des processus de catégorisation d'un sujet humain. De plus, une autre condition à satisfaire pour le psychomimétisme est de fournir des sorties facilement interprétables [Kan 96, ch. 2 et 3].

Un modèle cognitif pour la catégorisation

Le modèle cognitif que nous avons choisi de mettre en œuvre est issu des théories de Rosch [Ros 78b] et d'un modèle provenant de la psychologie de la décision, l'heuristique de la base mobile (HBM) proposée par Barthélemy et Mullet [Bar 86]. Les travaux de la psychologue Eleanor Rosch ont mis en évidence l'existence de *prototypes* utilisés par les sujets pour former les catégories. Le prototype était à l'origine un exemplaire de la catégorie, le meilleur représentant de celle-ci. Dès lors, la classification d'un objet s'effectue sur la base d'une similarité entre cet objet et le prototype de la catégorie. Depuis ces premiers travaux, Rosch a quelque peu modifié la notion de prototype : ce n'est plus une instance centrale mais distribuée (il peut y avoir plusieurs prototypes dans une même catégorie). De plus, les prototypes ne sont pas nécessairement des objets mais des abstractions d'un concept représentant la catégorie, concept défini par des *traits typiques* (valeurs sur certains des attributs qui décrivent les objets) [Ros 78a].

L'HBM permet de représenter le processus de décision d'appartenance d'un objet à une catégorie à l'aide d'une structure polynômiale comme ci-dessous :

$$P(C) = R^2F^4 + D^3T^2 + R^3$$

ce qui signifie que si un objet a été placé dans la catégorie C par le sujet, c'est parce qu'il vérifiait R=2 et F=4 ou D=3 et T=2 ou R=3, où D, T, R et F sont des attributs du langage de description des objets. Dans le cadre de la théorie de Rosch, chaque *monôme* (e.g. R^2F^4) s'interprète comme une *combinaison typique de traits*, qui caractérisent le prototype de la sous-catégorie des objets de C classés par ce monôme [Kan 96, ch. 1]. Un polynôme est donc la trace à la fois des structures

catégorielles (prototypes) et donc du processus de catégorisation, et du processus de classification (décision d'appartenance). Il s'interprète comme une règle de catégorisation dans le langage de description formé des valeurs possibles de tous les attributs décrivant les objets.

Les principes cognitifs sous-jacent à notre modèle sont au nombre de quatre. Les trois premiers sont issus de l'HBM : *parcimonie*, *fiabilité* et *décidabilité* [Bar 86] :

- *Principe de parcimonie* (ou d'*économie cognitive*) : en raison de son incapacité à traiter toutes les informations en même temps, le sujet extrait des groupes de traits dont la taille est suffisamment petite pour rester compatibles avec les capacités humaines de calcul et de rétention en mémoire à court terme (rationalité limitée).

- *Principe de fiabilité* : à première vue, ce principe va à l'encontre du premier. Le sujet, surtout s'il est expérimenté, sait qu'on attend de lui un jugement fiable. Concerné, aussi bien sur le plan personnel que social, par la fiabilité de ses décisions, il utilise un nombre suffisamment grand de traits qu'il compose de manière conjonctive, de façon à obtenir des critères pertinents pour lui.

- *Principe de décidabilité* Concerné par la nécessité de parvenir à une décision dans un nombre relativement important de situations, le sujet s'autorise à changer de critères, i.e. de structure de dominance, si les critères actuels ne lui permettent pas de décider.

Le quatrième principe est issu de l'Adaptive Resonance Theory de Grossberg [Car 87, Gro 82] et stipule qu'un objet sera classé dans une catégorie s'il entre en résonance avec un monôme représentatif de la catégorie, c'est à dire qu'il doit être suffisamment proche de celui-ci.

6.3.4 Le réseau Categ_ART

Le réseau Categ_ART [Kant, 1995, 1996] a été conçu pour mettre en œuvre ce modèle cognitif dans le cadre du connexionnisme psychomimétique. Son architecture est analogue au réseau ART [Car 87] : c'est une architecture d'une réseau à prototypes, composé de deux couches F_1 et F_2, comme on le voit sur la figure 6.11 ci dessous.

Trois sous-systèmes composent Categ_ART : *attention*, *orientation* et *agrégation*. Le sous-système d'attention est formé par les couches F_1 et F_2. Les cellules de la couche F_1 reçoivent en entrée les traits définissant les objets, la cellule n^o i codant la

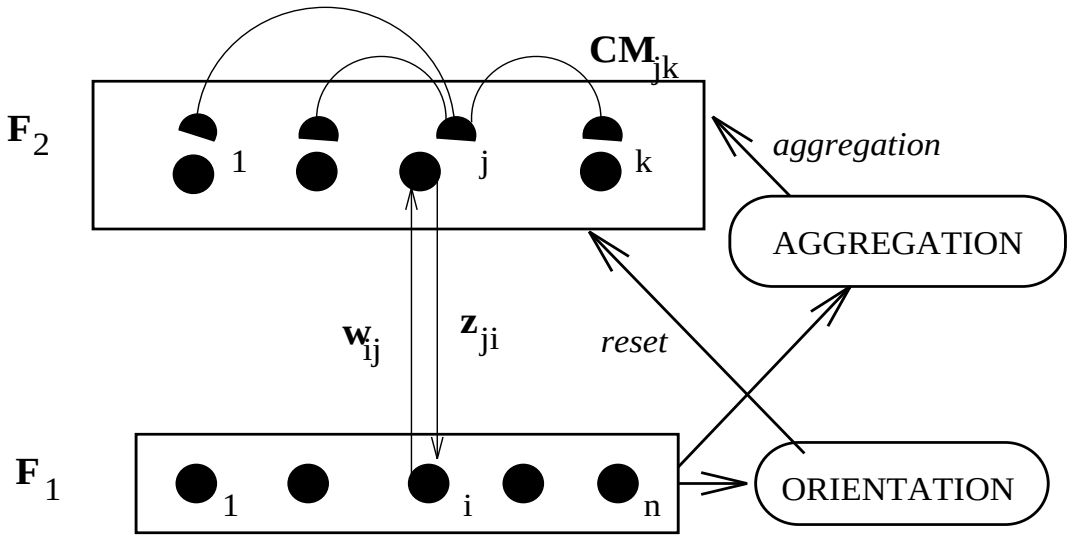


FIG. 6.11 – *Architecture du réseau Categ_ART*

valeur discrète de l'attribut n^o i de l'objet. Les cellules de la couche F_2 représentent les monômes qui servent à catégoriser les objets. Les traits qui définissent le monôme associé à la cellule j sont codés par les connexions montantes w_{ij}. Les monômes ne sont pas prédéfinis, ils sont calculés dynamiquement suivant les objets présentés en entrée du réseau. Une catégorie étant définie par une disjonction de monômes, il nous faut un moyen de regrouper les monômes formés. Ceci est assuré par les connexions CM, entre les cellules de F_2 : une valeur positive (excitatrice) de CM_{jk} indique que les monômes j et k sont dans une même catégorie ; si CM_{jk} est négative (inhibitrice), cela signifie que les 2 monômes sont dans des catégories différentes. Notons que ces connexions n'existent pas dans le réseau ART, car celui-ci associe une catégorie à un seul monôme.

L'algorithme d'apprentissage de Categ_ART est présenté en détail dans [Kan 96, ch. 5], dans le cas où les données sont à *valeurs discrètes*. Il se divise en *trois étapes*. La première est une application directe du *principe de parcimonie*. Son but est de former des monômes composés d'un seul trait. Lorsqu'un objet est présenté à l'entrée du réseau, chacun des traits qui le code donne naissance à un monôme en F_2. De plus, l'objet produit, par l'intermédiaire des *connexions montantes* w, une sortie

sur les cellules de F_2 déjà créées : un critère de résonance est appliqué, utilisant les *connexions descendantes* z qui jouent le rôle de seuils. L'objet est classé dans toutes les cellules avec lesquelles il entre en résonance, à condition de placer deux objets dans une même catégorie uniquement si le sujet dont on veut calculer les règles de catégorisation a fait de même. Cette étape est répétée pour tous les objets.

La seconde étape applique le *principe de fiabilité*. Nous définissons pour chaque monôme un *taux d'erreur de catégorisation*. Lorsque ce taux est supérieur à un seuil η donné, le sous-système d'agrégation est activé, afin d'ajouter un trait au monôme pour diminuer le taux d'erreur. Ce processus se répète jusqu'à ce que tous les monômes soient *acceptables*, i.e. qu'ils aient un taux d'erreur inférieur à η.

La troisième étape vise à regrouper les monômes pour former les catégories proprement dites, conformément au *principe de décidabilité*. Pour cela, tous les objets sont présentés au réseau et interagissent avec les monômes de la couche F_2, permettant de calculer les connexions inter-monômes CM. Les équations d'évolution de CM sont telles qu'à la fin de la présentation, les connexions entre les monômes sont positives pour des monômes de même catégorie, négatives pour des monômes de catégorie différentes.

Évaluation de Categ_ART

Le réseau Categ_ART est le seul des 3 SCA que nous avons présentés qui réalise une interprétation corrélée, puisque le réseau construit pendant l'apprentissage les règles qui permettront d'expliquer les classifications. Ceci est notamment dû au choix du modèle cognitif fondé sur l'HBM, qui permet de voir le processus de catégorisation également comme un processus de décision : ces deux opérations (catégorisation et classification) sont intégrées dans un seul modèle et réalisées conjointement par Categ_ART, ce qui est une propriété intéressante d'un point de vue théorique en modélisation cognitive. Le pouvoir explicatif de Categ_ART est donc très bon.

De plus, les expériences que nous avons menées ont montré que le pouvoir de Categ_ART était au niveau du perceptron multi-couches et des arbres de décision. Categ_ART possède l'avantage sur ces derniers (C4.5, OC1) de fournir, pour un même taux de succès en généralisation, des règles nettement plus courtes et donc plus lisibles (deux fois plus courtes que C4.5, trois fois plus courtes qu'OC1) [Kan 96, ch. 6].

Bibliographie

[Abd 94] H. Abdi. *Les réseaux de neurones.* Sciences et technologies de la connaissance. Presses Universitaires de Grenoble, Grenoble, 1994.

[Bar 86] J.-P. Barthélemy et E. Mullet. Choice basis: a model for multi-attribute preferences. *British Journal of Mathematical and Statistical Psychology*, 43:106–124, 1986.

[Boc 89] L. Bochereau et P. Bourgine. Implémentation et extraction de traits sémantiques sur un réseau neuro-mimétique: exemple de la première annonce au bridge. *Actes du congrés Neuro Nîmes*, 1989.

[Boc 91] L. Bochereau, P. Bourgine, et G. Deffuant. Equivalences entre classificateurs connexionnistes et classificateurs logiques. *Intellectica*, (12):139–158, 1991.

[Bre 84] L. Breiman, J.H. Friedman, R.A Olshen, et C.J. Stone. *Classification and regression trees.* Wadswoth, Belmont, CA, 1984.

[Car 87] G. Carpenter et S. Grossberg. A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer vision, Graphics and Image Processing*, 37:54–115, 1987.

[Cyb 89] G. Cybenko. Approximation by superpositions of a sogmoidal function. *Mathematics of Control, Signals and Systems*, 2:303–314, 1989.

[Did 82] E. Diday, J. Lemaire, J. Pouget, et F. Testu. *Eléments d'analyse de données.* Dunod, 1982.

[Gro 82] S. Grossberg, éditeur. *Studies in Mind and Brain.* Reidel, Boston, 1982.

[Kan 96] J.-D. Kant. *Modélisation et mise en œuvre de processus cognitifs de catégorisation à l'aide d'un réseau connexionniste.* Thèse de doctorat, Université de Rennes I, Janvier 1996.

[LC 86] Y. Le Cun. Learning process in an asymmetric threshold network. E. Bienenstock, F. Fogelman-Soulié, et G. Weisbuch, éditeurs, *Disordered systems and biological organization*, NATO Series. Springer Verlag, Berlin, 1986.

[Qui 93] J.R. Quinlan. *C4.5: programs for machine learning.* Morgann Kaufmann, San Mateo, CA, 1993.

[Ros 58] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization of the brain. *Psychological Review*, (65):386–408, 1958.

[Ros 78a] E. Rosch. Principles of categorization. E. Rosch et B. B. Lloyd, éditeurs, *Cognition and categorization.* Erlbaum, Hillsdale (N.J.), 1978.

[Ros 78b] E. Rosch et B. B. Lloyd, éditeurs. *Cognition and categorization.* Erlbaum, Hillsdale (N.J.), 1978.

[Rum 86] D. E. Rumelhart, G. E. Hinton, et R. J. Williams. Learning internal representations by error propagation. D. E. Rumelhart et J. L. Mc Clelland, éditeurs, *Parallel distributed processing*, chapitre 8. MIT Press, Cambridge, 1986.

[Wid 60] B. Widrow et M.E. Hoff. Adaptive switching circuits. 1553-1, Stanford University, Stanford, USA, 1960.

Unit´e de recherche INRIA Lorraine, Technop^ole de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unit´e de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unit´e de recherche INRIA Rh^one-Alpes, 46 avenue F´elix Viallet, 38031 GRENOBLE Cedex 1
Unit´e de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unit´e de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399